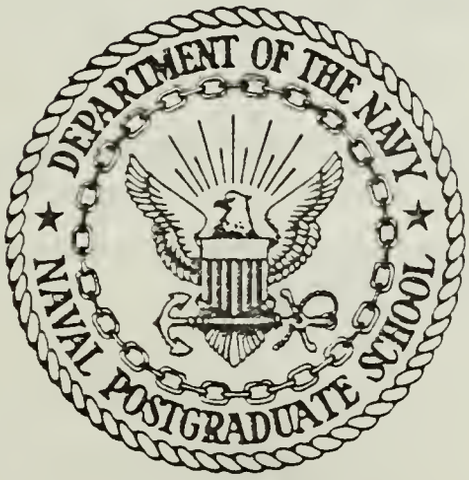
Application of numerical optimization in modern control.

Olson, Thomas F.

NAVAL POSTGRADUATE SCHOOL
Monterey, California



THESIS

APPLICATION OF
NUMERICAL OPTIMIZATION
IN MODERN CONTROL

by

Thomas F. Olson

December 1986

Thesis Advisor R.H. Nunn

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution is unlimited
2b DECLASSIFICATION / DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S)	5 MONITORING ORGANIZATION REPORT NUMBER(S)
6a NAME OF PERFORMING ORGANIZATION Naval Postgraduate School 6b OFFICE SYMBOL (If applicable) 69	7a NAME OF MONITORING ORGANIZATION Naval Postgraduate School
6c ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000	7b ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000
8a NAME OF FUNDING / SPONSORING ORGANIZATION 8b OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c ADDRESS (City, State, and ZIP Code)	10 SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO / PROJECT NO / TASK NO / WORK UNIT ACCESSION NO

11 TITLE (Include Security Classification)

APPLICATION OF NUMERICAL OPTIMIZATION IN MODERN CONTROL

12 PERSONAL AUTHOR(S)
Olson, Thomas F.

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
Master's Thesis	FROM _____ TO _____	1986 December	147

16 SUPPLEMENTARY NOTATION

17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Optimal Control, Numerical Optimization, Controller Design, Performance Index, Numerical Methods

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

The theory of optimal control presents a powerful method of controller design. The basis of the method is selection of a "performance index" which compares the actual to desired controller performance for such items as output response and energy consumption. In most previous work, the design process has radically simplified the performance index for analytic feasibility and practical utility. In order to provide a more accurate and versatile method, state of the art numerical optimization methods, using the Automated Design Synthesis Program, are applied to numerical modeling of multivariable controllers, using the Dynamic Simulation Language. The resulting optimum values stemming from the analysis are the controller gains which minimize the desired performance index for a specified set of system constraints.

20 DISTRIBUTION / AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
22a NAME OF RESPONSIBLE INDIVIDUAL Nunn, Robert H.	22b TELEPHONE (Include Area Code) 408-646-2365 22c OFFICE SYMBOL 69Nn

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted
All other editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE

Approved for public release; distribution is unlimited.

Application of Numerical Optimization in Modern Control

by

Thomas F. Olson
Lieutenant Commander, United States Navy
B.S., Bradley University, 1973

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
December 1986

ABSTRACT

The theory of optimal control presents a powerful method of controller design. The basis of the method is selection of a "performance index" which compares the actual to desired controller performance for such items as output response and energy consumption. In most previous work, the design process has radically simplified the performance index for analytic feasibility and practical utility. In order to provide a more accurate and versatile method, state of the art numerical optimization methods, using the Automated Design Synthesis Program, are applied to numerical modeling of multivariable controllers, using the Dynamic Simulation Language. The resulting optimum values stemming from the analysis are the controller gains which minimize the desired performance index for a specified set of system constraints.

THESIS DISCLAIMER

The reader is cautioned that computer programs developed in this research may not have been exercised for all cases of interest. While every effort has been made, within the time available, to ensure that the programs are free of computational and logic errors, they cannot be considered validated. Any application of these programs without additional verification is at the risk of the user.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

The author would like to extend his deepest thanks to Professor Robert H. Nunn for his continual nurturing and enthusiasm and to Associate Professor David L. Smith for his numerous suggestions and support.

The author would be remiss in failing to extend a word of thanks to Associate Professor David Salinas for his guidance with optimization techniques, and the W.R. Church Computer Center Staff personnel Patricia Collins and June Favorite for their assistance in the linking of the ADS and DSL/VS computer programs.

The author wishes to extend a special note of appreciation to his wife and son for their moral support and understanding in the many hours he spent away from them and in front of various computer terminals.

I. INTRODUCTION

The design of controllers has been an engineering endeavor since the 1920's. The road to success was significantly changed by the introduction of the International Business Machine Model 360 computer in the 1960's, which allowed the control designer to develop numerical models for various systems and effieiently test his design before prototyping. The development of FORTRAN programs such as the Continuous System Modeling Program (CSMP), and Dynamic Simulation Language (DSL) greatly simplified the modeling proeess and allowed the designer to have highly accurate numerieal tools at hand, rather than having to rely on elosed form or reduced-matrix type solutions such as the Riccatti equation.

Along similar lines the engineering discipline of optimization has developed. Numerical optimization, using mainframe and micro computers, has rapidly developed over the past ten years with the work of G. N. Vanderplaats and others. His FORTRAN programs Control Program for Engineering Synthesis (COPES) and most recently Automated Design Synthesis (ADS) have provided routines that implement all of the known methods available to do numerical optimization.

The goal of optimization has been always to provide the best for the least, and this credo ean be applied to the design of all controllers. One method of aceomplishing this task will be demonstrated by the author. It involves the modeling of single input, single output (SISO) or multiple input, multiple output (MIMO) systems using DSL, specifying an applicable performance index by which to judge the system, defining constraints which specify the desired system performance eharacteristics as well as verifying system stablity, and then ealling upon ADS. Through iterations the optimization routine finds the necessary values for system gains (defined as design variables) whieh will minimize the performance index (defined as the objeetive function). The final result being the "best" eontroller design.

II. MODERN CONTROLLER DESIGN

A. ANALYSIS TECHNIQUES

Two general methods of controller design have developed, each having certain advantages, depending upon the application of the controller involved. The first is frequency domain analysis, in which the response of the desired system to a sinusoidal input is examined. The other method is time domain analysis, where the system response to various time dependent inputs such as impulse, step, or ramp functions are investigated using a state-space representation of the system. This second method lends itself more easily to numerical modeling techniques that can be programmed on a computer, and will be the method of choice in this investigation.

B. NUMERICAL ANALYSIS TECHNIQUES

1. CSMP

The CSMP-III program was developed by International Business Machine Corporation (IBM) to provide users with a simplified method for numerically solving a system of ordinary differential equations and related mathematical problems. [Ref. 1:p. 2]. Today it is a widely used and respected tool for the design of control systems.

Some drawbacks to CSMP were found when the initial considerations were made on how to perform numerical controller optimization. The most significant areas of concern were that CSMP uses the FORTRAN/HX complier, it is only available locally in the Multiple Virtual System (MVS) environment, it uses single precision variables throughout, and that graphics are available only after processing through the DISSPLA graphics system. These facts led to a decision to select a different method to do controller optimization.

2. DSL/VM

DSL/VM is a recent product from IBM [Ref. 2:p. iii]. The program is similar to CSMP in that simplified source code is translated into standard FORTRAN statements (as output to the file FORT FORTRAN), compiled and then run in order to perform the desired system simulation. Following the successful run, tabled output is directed to the user along with any desired graphic displays. Fig. 2.1 is a flowchart of this procedure.

Significant improvements from CSMP to DSL include use of the FORTRAN/VS compiler, Virtual Machine (VM) or MVS environment availability, on line graphics capability, double precision calculations throughout, and automatic linkage to user defined libraries. Additionally, a one time job initialization segment is available which is separate from the model initial segment. Thus the programmer can easily rerun the time dependent simulation for the the same model, after changing any desired parameters, within the same job specification. These facts were significant reasons for selecting DSL as the modeling tool to use for controller optimization.

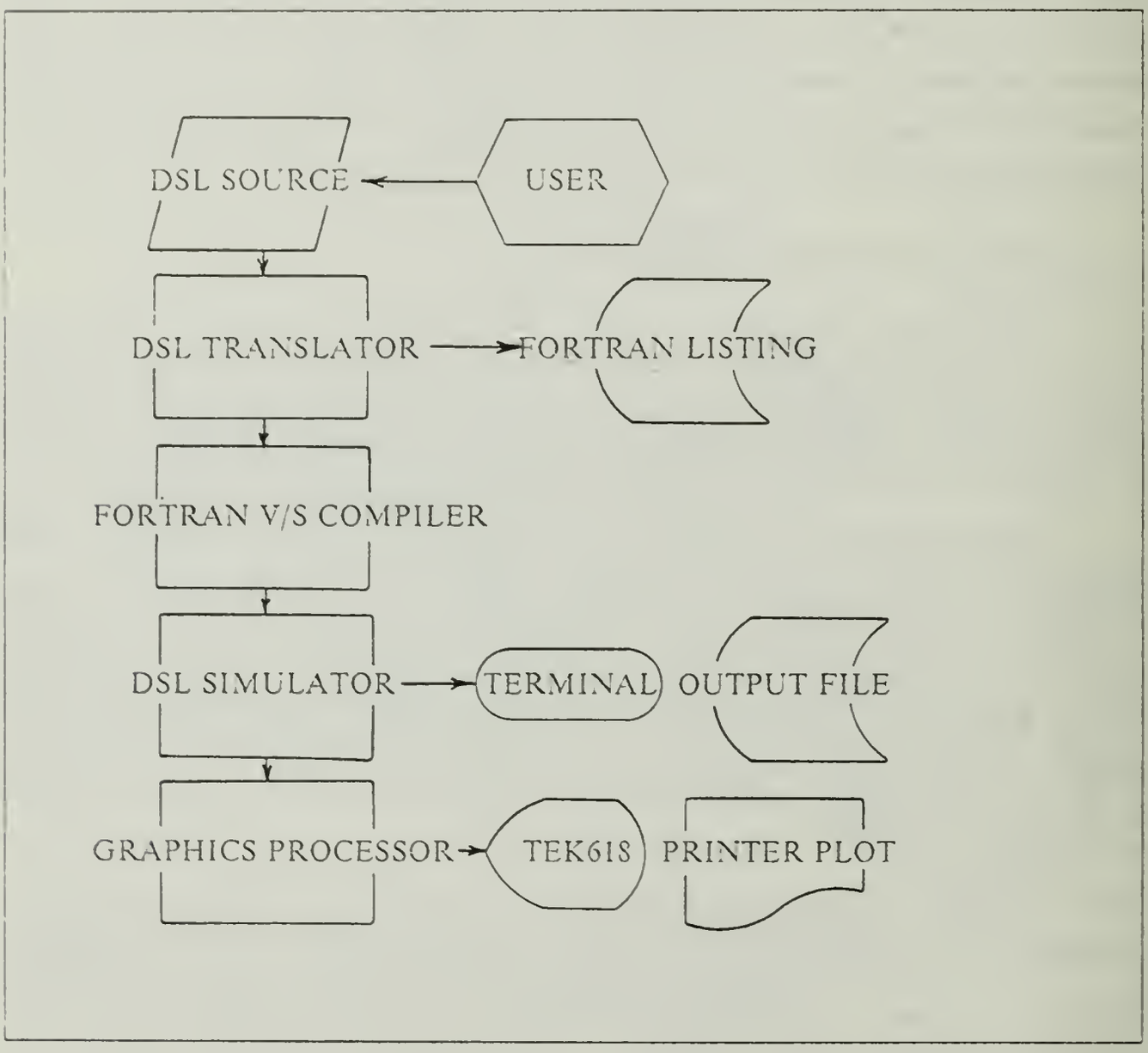


Figure 2.1 DSL/VS Flow Chart

III. OPTIMIZATION

A. ANALYSIS TECHNIQUES

1. Basic Methods

The determination of the "best" value for any design requires that several items be considered. One is the resource that can be varied, made up of what are called the design variables. Also included are the mathematical relationships that describe how the resource relates to the design environment. These relationships define what are called the objective function and constraint functions. Many diverse methods have been developed to determine the values of the design variables that cause the objective function to be at its optimum value. With the advent of the computer, numerical computation methods have become the most practical ones for treatment of complex systems, supplanting graphical and analytic determinations.

2. Numerical Methods

Many different approaches have been developed for solving the general optimization problem by numerical methods. *Numerical Optimization Techniques for Engineering Design* [Ref. 3] describes most of the common analytic techniques and the corresponding algorithms to perform this task using a computer. A major extension of this work was the development of the Control Program for Engineering Synthesis (COPES) and Constrained Minimization (CONMIN) FORTRAN routines that accomplish the various numerical tasks necessary to successfully determine an optimal solution to problems that can be appropriately defined.

a. COPES and CONMIN Programs

COPES is a FORTRAN program that was developed as a method for easily coding the parameters required to define an optimization problem for solution by a computer. Its use requires that a FORTRAN subroutine called ANALIZ be written which evaluates the objective function and all constraint functions. Previously, the FORTRAN subroutines grouped under the name CONMIN were then called to conduct the desired optimization analysis. Recent improvements to the CONMIN routines have been generated and the package has been renamed Automated Design Synthesis Program (ADS) [Ref. 4]. To use COPES with ADS, the routine COPESA should be called.

A major drawback to the use of COPESA is the specific formatting and subroutine requirements. This hinders interactive optimization, and has resulted in the selection of an alternative method for performing controller optimization.

b. ADS Program

ADS can be called directly from any FORTRAN program [Ref. 4:pp. 5-8]. Prior to making the call, all input parameters must be defined, and a looping logic must be provided in the calling program that causes the objective and constraint functions to be evaluated upon the completion of each optimization iteration, as depicted in Fig. 3.1 .

With this simple logic many engineering problems can benefit from the advanced capabilities offered by the ADS program. The specific area that has served as a focus for this study is the interfacing of controller design, done with DSL, with optimization using ADS.

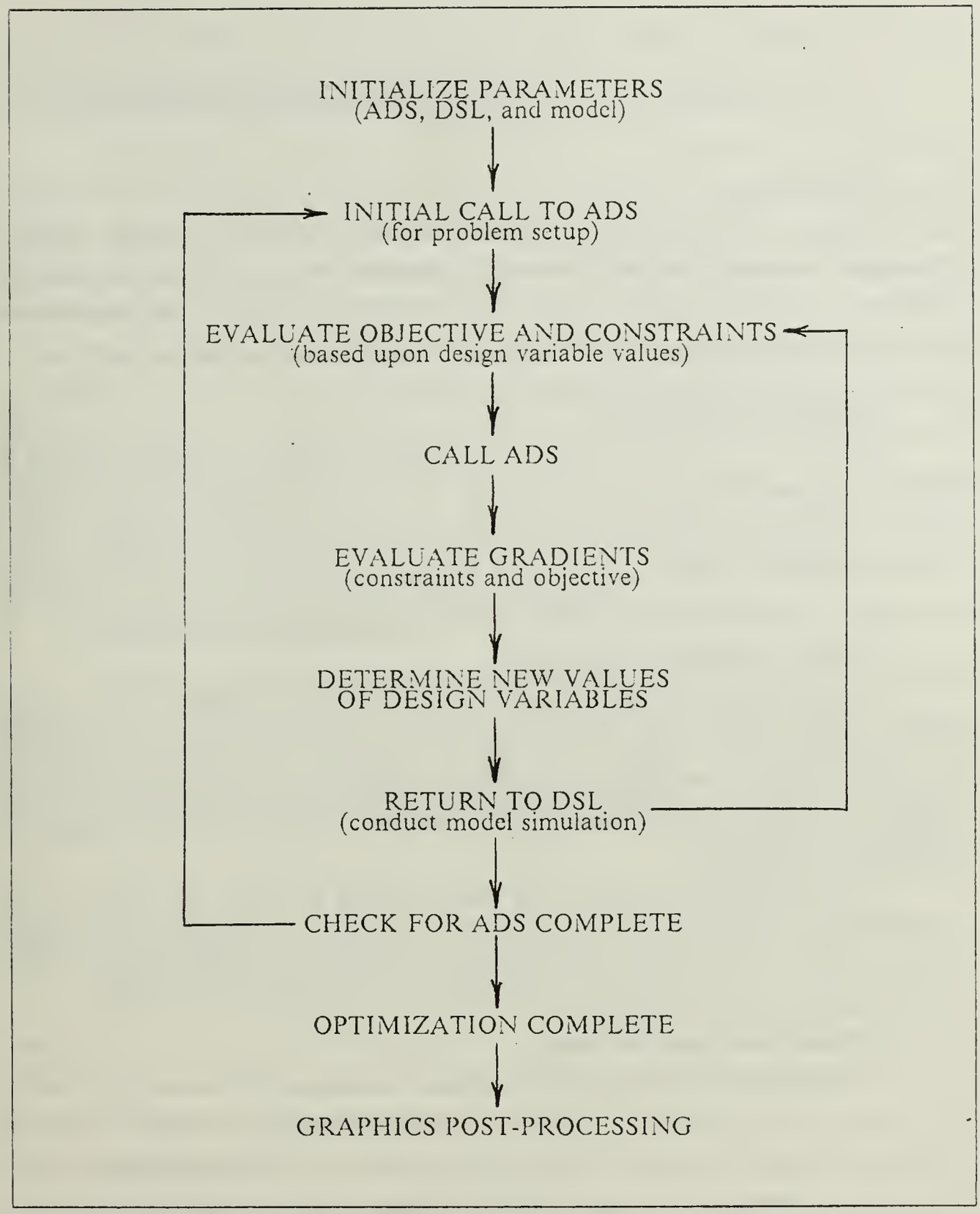


Figure 3.1 Program Logic Calling ADS Directly

IV. AUTOMATED DESIGN SYNTHESIS/DYNAMIC SIMULATION LANGUAGE

A. OVERVIEW

Linking of the Automated Design Synthesis Program and the Dynamic Simulation Language has been successfully demonstrated, and the acronym ADSL represents the close coupling of these two FORTRAN programs. In the following discussion the methodology of how these two programs interface will be explained.

B. PROGRAM SEGMENTS

Figure 4.1 is a flowchart of how the individual segments of a DSL program are utilized with the ADS optimization code. The *DSL Language Reference Manual* [Ref. 2] is the source document on the syntax and logic of the DSL language. Chapter 3 of the manual explains in detail the function of each phase of DSL program processing. For the proposed method of controller design each phase is utilized.

1. Program Initialization (INITLZ)

This portion of a DSL program is the first executed segment of code. In it arrays are dimensioned, non-standard variable types (fixed, complex, etc.) are defined, and initial values of variables are assigned. Additionally, DSL parameters which specify the output variables, the desired integration method, and the technique for plotting can be specified.

The important fact about this section is that it is only executed one time. Thus the initial values that ADS requires about the status of the optimization, denoted by the variable INFO, and the array dimensioning can be done in this section.

2. Run Initialization (INITIAL)

The beginning of each system response simulation starts at this point with a call to ADS. On the initial call, ADS is provided with best-guess values for the design variables, the upper and lower bounds for each design variable, along with the number and types of constraints and their allowable ranges. ADS will return new values for the design variables to be used in the controller design. On subsequent calls to ADS updated values of the objective and constraint functions will be provided to ADS for its determination of the optimal values of the design variables.

The remainder of this program section sets various constants at the initial values that will be needed later in the dynamic section.

3. Derivative Section (DERIVATIVE)

This section is the heart of the simulation. The inputs to the system are defined, the system transfer function is specified, and various outputs, states and the objective function are evaluated. Great latitude is possible in each of these specifications, and for this reason DSL is an outstanding method for analysis of controller design.

This section also determines the integration time step that the simulation will take, based upon two criteria. The first is how much time is remaining before a specified data output time is reached, and the second is how big an integration step can be taken while maintaining the specified relative and absolute errors. The output time is determined through the use of the DELPRT and DELPLT commands. These control parameters indicate the desired time intervals for saving intermediate values of specified variables for both plotting and listing purposes.

4. Dynamic Section (DYNAMIC)

Once the derivative section finalizes what the size of the next integration interval will be, it then calculates the magnitude of the various system states, outputs, and objective function at the end of the integration interval. This information is then passed onto the dynamic section where it can be evaluated to determine the various constraint values and whether or not the simulation is complete (specific information on the evaluation of these parameters will be left to subsequent discussions of the single input, single output (SISO) example). If the simulation is incomplete, computational flow is is directed back to the derivative secton.

When the completion of the simulation run is determined, by either reaching a specified slope of the objective function, or a maximum time limit, the DSL routine ENDRUN is called. This procedure completes all outstanding calculations and then transfers control to the terminal segment.

5. Terminal Section

In this last program segment the ADS status variable INFO is checked to determine whether or not the optimization is complete. If it is, the DSL program is terminated by calling the routine ENDJOB. If further optimization is required, control is transferred back to the initial segment.

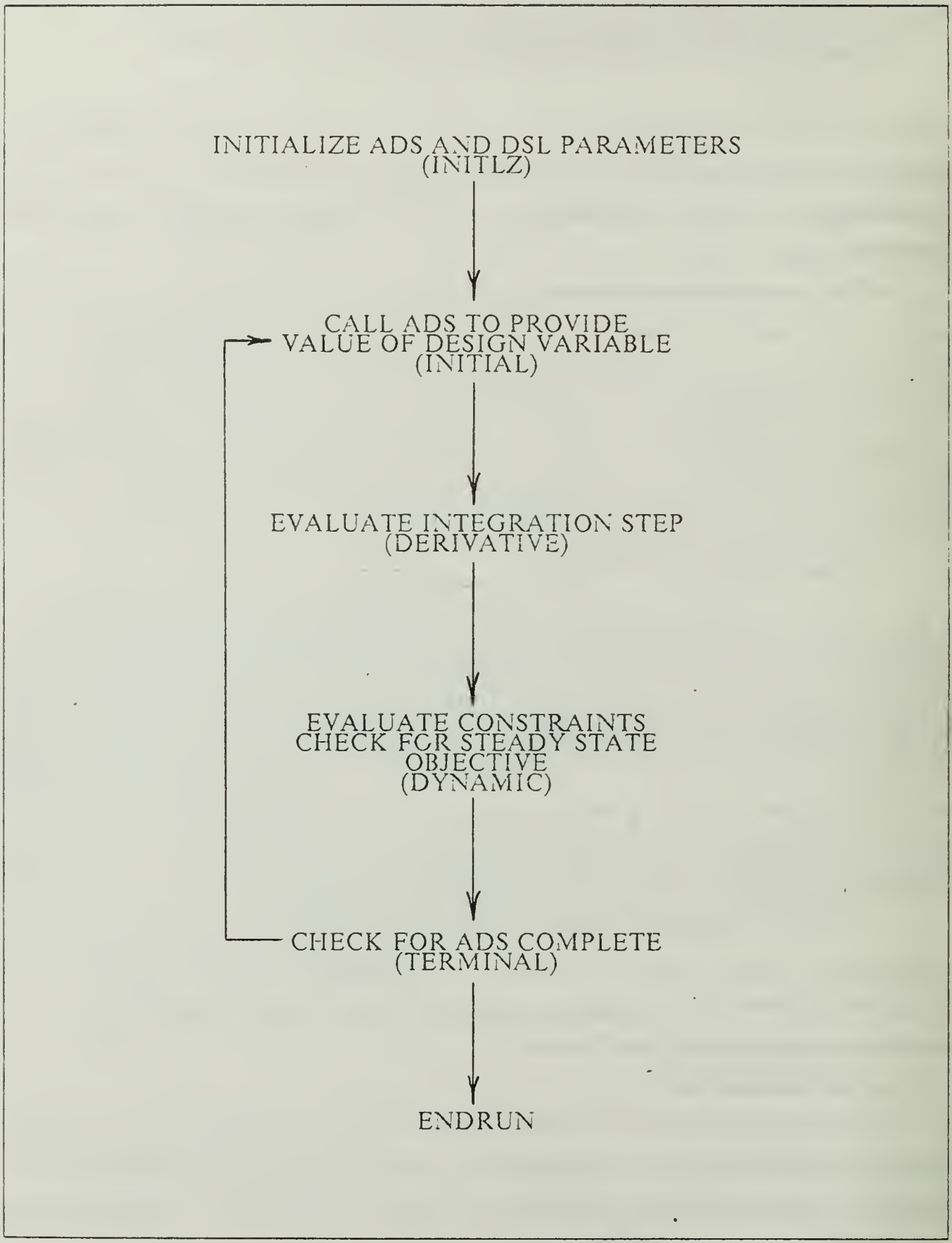


Figure 4.1 ADSL Program Segments

V. SINGLE INPUT SINGLE OUTPUT PROBLEM ANALYSIS

A. CASE 1 SECOND ORDER SYSTEM

In order to obtain a baseline for the ADSL program the well behaved, general, second order system was taken for examination. Ogata [Ref. 5] completely describes this simple system shown in Fig. 5.1 and its responses to various inputs. For simplicity this system will be called Case 1. In the present study, the input is taken to be the unit step function, R(s)= 1.0 for time greater than zero, and the magnitude ω was set equal to 1.0.

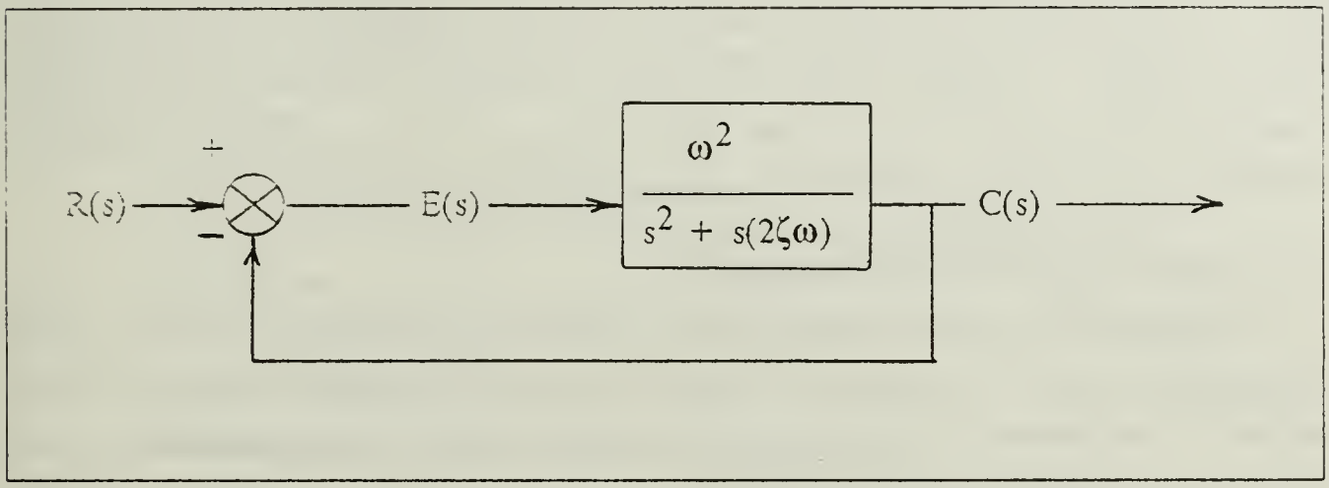


Figure 5.1 Case 1 Block Diagram

Various analytic optimization methods have also been developed [Ref. 5:pp. 296-306] for this system that can be used to verify the ADSL program.

Objectives of this baseline investigation were as follows:

- Determine the most efficient and correct integration method.
- Determine the most efficient optimization strategy.
- Determine the variations to be expected from using different performance indices.
- Determine the most overall-efficient method to obtain optimum design variable values.

A sample program to accomplish these tasks is listed in Appendix C.

B. PROBLEM FORMULATION

1. Objective Function

The fundamental part of any optimization problem is definition of an objective function. For this problem the performance index is the logical relationship that defines how "good" a system is. Ogata [Ref. 5:p. 296], defines the various performance indices that have been used in this investigation, and these will be described in the following discussion.

2. Design variable

The design variable selected for this investigation was the damping ratio (ζ). It is a reasonable parameter because it directly effects how the system responds to any input over time. Thus if ζ is increased, the system output response will be slower and a larger accumulated error between the actual to desired output may be generated with respect to time. This larger error could then increase the value of the performance index/objective function. On the other hand, insufficient damping will lead to excessive output oscillations which also tend to increase the objective function.

C. APPLICATION OF CONSTRAINTS

Ogata [Ref. 5:p. 232] defines transient response specifications for the typical second order system. In almost all cases, these standard definitions can be directly applied to the response of a DSL model. A brief review of these specifications and how they are applied are discussed below.

1. Maximum Overshoot

How much the output response of an under-damped system goes beyond that which is desired, is the maximum overshoot. The larger this value is, the more energy is wasted, as a restoring force must be applied to return the system to the desired state. In addition, physical systems may be limited in the amount of amplitude overshoot that can be tolerated. A simple constraint of the maximum allowable amplitude, as a fraction of the intended steady state, can be defined.

2. Delay Time

The time the system requires to reach fifty percent of its intended response is the system delay time. It indicates how fast the system begins to respond, and a constraint of a maximum time can be established.

3. Rise Time

When the system initially reaches 100 percent of the desired response, the rise time has been reached. Like the delay time, this parameter is also an indication of how quickly the system will respond. For under-damped systems, a constraint of maximum rise time can be programmed.

4. Peak Time

The time of maximum response is called the peak time. It also shows how fast the system responds. In cases where a peak time occurs, a maximum peak time constraint can be specified.

5. Settling Time/Completion Time

Settling time is defined [Ref. 5:p. 236] as when the response falls and stays within a prescribed percentage (typically two or five percent) of the total response. For computational considerations, the settling time was redefined for the two possible response conditions:

1) Underdamped case: that time when the absolute maximum of a response oscillation peak is within one percent of the total response transient.
2) Overdamped case: that time, greater than five time constants, at which the absolute system response is within one percent of the total response. A time constant is defined as that time required for the system output response to reach sixty three percent of the final response value.

In order to keep the above definition separate from settling time, the new consideration will be called completion time.

The different constraint criteria are illustrated in Fig. 5.2, with the constraints of maximum amplitude, rise time, delay time, peak time, and completion time, being designated by (1) to (5) repectively.

D. INTEGRATION CONSIDERATIONS

DSL provides nine optional integration methods which can be grouped into the following types:

- Fixed-step
- Variable-step
- Variable-step, Variable-order.

The default method is the Runge-Kutta Fifth Order variable-step method. Selection of the method which is best suited to the controller design problem is discussed below.

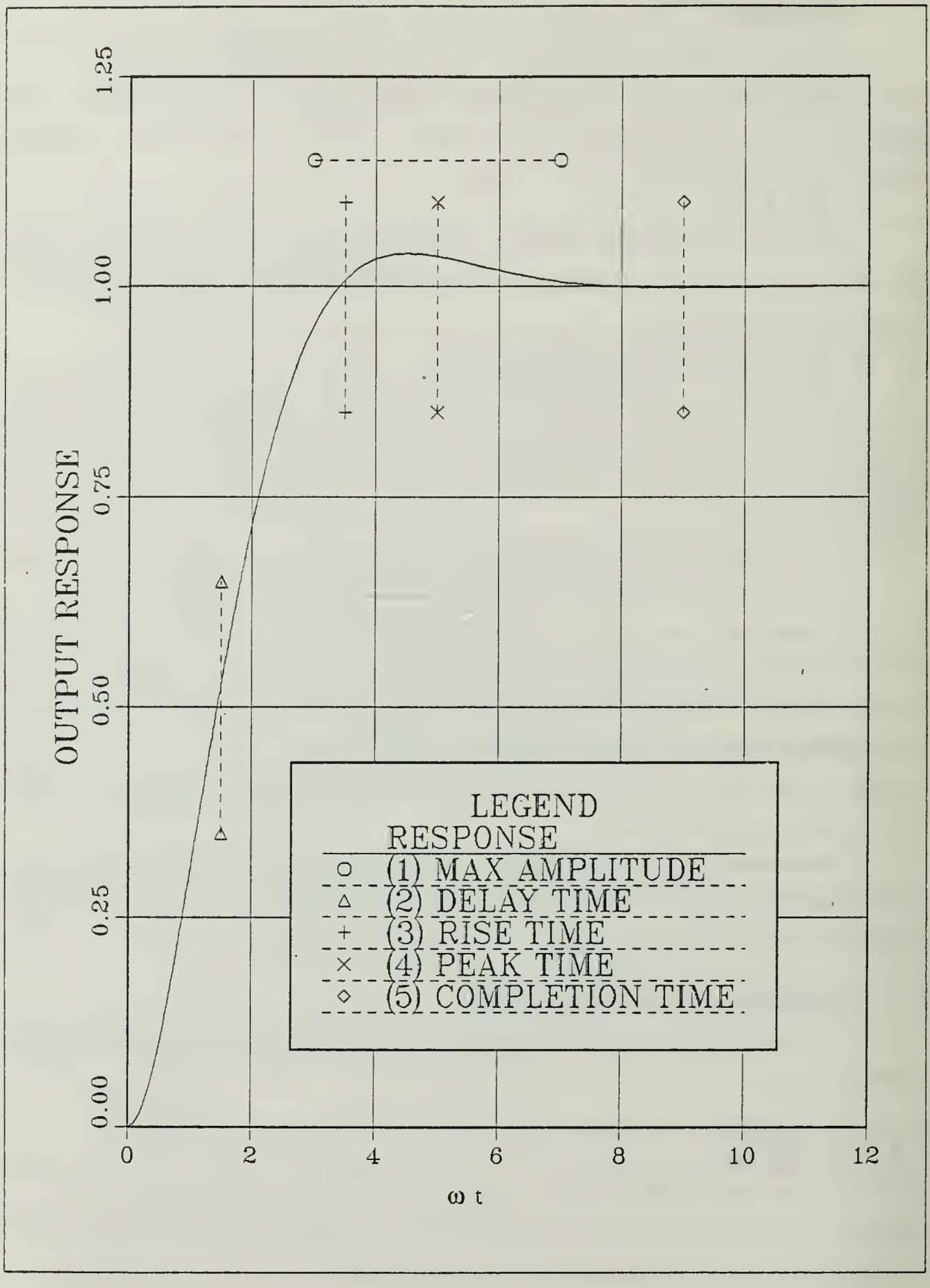


Figure 5.2 Case 1 Demonstration of Constraints

1. **Limits of Integration Determination**

For this analysis the objective function has been defined as the system performance index. The particular performance indices to be examined are weighted time integral errors between the actual and desired output response of the system from the time the input is applied to infinity. Noting that infinity is an impractical number for a computer (or anything else, for that matter), a reasonable approximation must be found. Two criteria were considered for determining what cut-off time to use to mark the end of calculations, or time equivalent to infinity.

One cut-off time criteria would be to simply select the completion time, but this has a significant drawback. Different values of the design variable will cause different cut-off times, and thus this method does not provide a consistent manner for evaluating the objective function for different designs. A second method is to evaluate the rate of growth of the objective function with time. For a stable system the objective function initially increases rapidly due to the mis-match of input and output. As time progresses this error approaches zero and the rate of change of the objective function slows down. Once the completion time is reached, if the rate of change of the objective function is evaluated to be essentially zero, a consistent method of determining the cut-off time can be established for all values of the design variable.

2. **Integration Method Analysis**

Typical output and objective functions are shown in Figs. 5.2 and 5.3. These rapidly increase from zero and approach a constant value asymptotically. This type of behavior is described by Speckhard [Ref. 1:p. 268], and is characteristic of "stiff" systems. In considering which integration method would best deal with this stiff system problem, fixed-step methods were ruled out due to their inability to accurately deal with rapidly changing functions. The Runge-Kutta Fifth Order (RK5), Adams variable-order (ADAMS), and Gear Full Jacobian (STIFF) variable-step methods were compared to determine which would result in the maximum computational efficiency.

The computational efficiency in this problem can be considered in two parts:

1) The number of individual integration steps
2) The number of objective function evaluations requested by the optimization routines for gradient determination.

The conclusion is then to use an integration method that maximizes the step size and an optimization routine that minimizes the number of function evaluations. The *DSL Reference Manual* [Ref. 2:p. 3.16] describes the reasons for differences in execution time

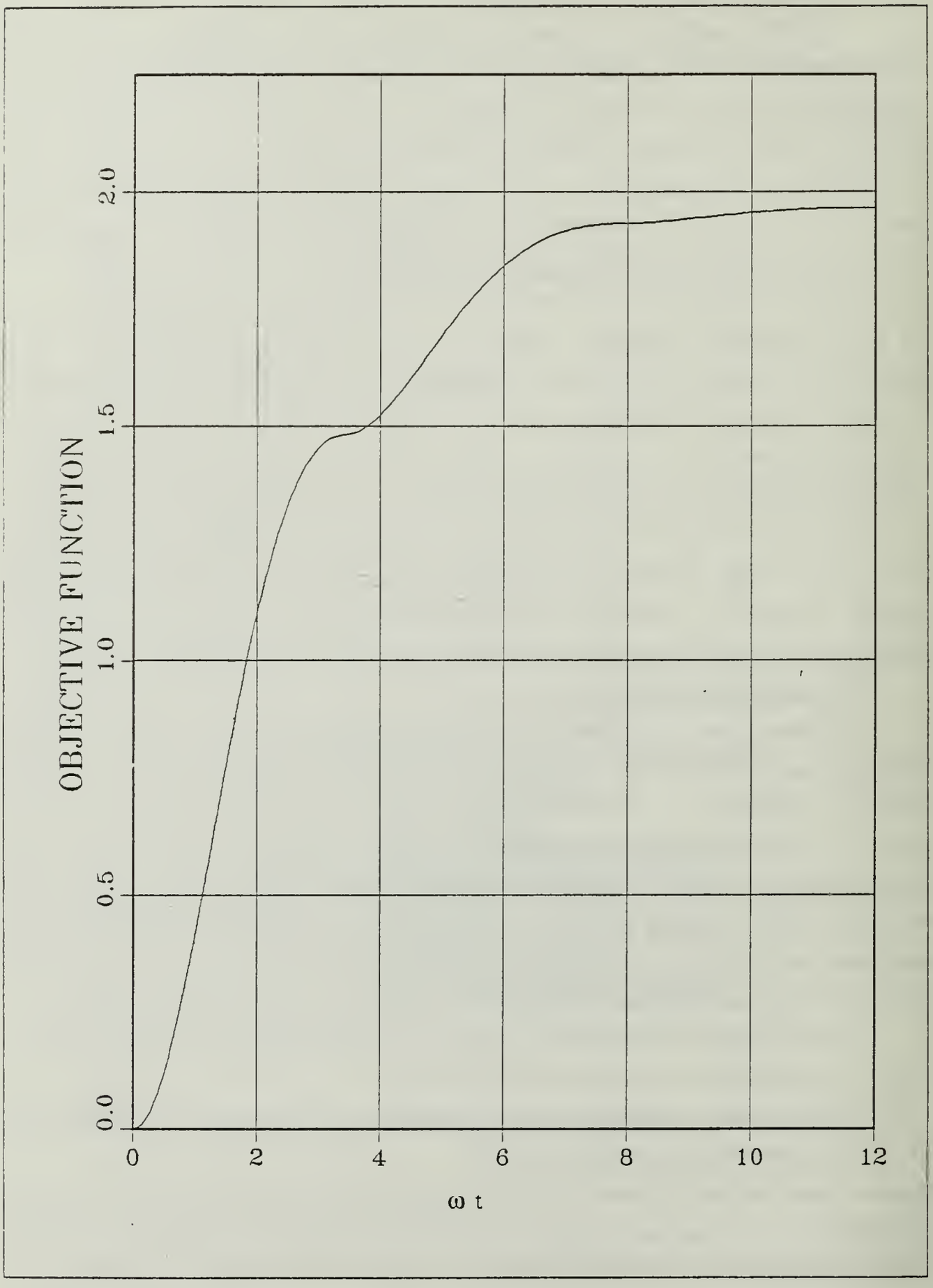


Figure 5.3 Case 1 Objective Function

between the integration methods, and points out that for oscillatory solutions, such as those seen for the second order underdamped system response, all three of the methods are essentially equal. Consequently, the best method to use is the one that generates the least number of objective function evaluation requests by the optimizer.

Three ADSL programs were generated, each using a different integration method, but using the same input, system, optimization method, and objective function (performance index). Discussion of optimization methods and performance indices will follow. In the problem at hand the optimization method was to use an exterior penalty function strategy with the Broydon - Fletcher - Golfarb - Shanno optimizer and polynomial interpolation for the one dimensional search. For the performance index the integral square error (ISE) was employed. The results are compared in Table 1. The applied constraints, designated by (1) to (5) repectively in Fig. 5.2 and Table 1, were:

(1) maximum amplitude
(2) rise time
(3) delay time
(4) peak time
(5) completion time

Interesting points from this investigation include:

- All three integration methods found the same result within the default tolerance of ADS.
- Each method, starting with the same initial design variable value, calculated a different initial objective function value.
- The number of objective function calls for each method was almost the same, but the STIFF method, as one might have predicted, required the fewest.

E. OPTIMIZATION METHODS

In order to use ADS, an optimizer and one-dimensional search must be selected. Optionally, a strategy may be invoked.

1. Optimization Strategy

An optimization strategy is simply the algorithm by which the overall optimization problem is transformed into one in which constraints are weighted or eliminated and the objective function may be redefined by use of a "pseudo-objective" function. Although use of a strategy is not required, some problems have shown significant computational improvement as a consequence.

TABLE 1

CASE 1 INTEGRATION METHOD RESULTS

Method	*RK5*	*ADAMS*	*STIFF*
Initial Design	1.0	1.0	1.0
Initial Objective	4008.9	12334.	5073.9
Final Design	0.717	0.718	0.720
Final Objective	1.968	1.975	1.965
Constraint 1	-.361	-.361	-.362
Constraint 2	-.157	-.190	-.157
Constraint 3	-.121	-.103	-.077
Constraint 4	-.170	-.113	-.016
Constraint 5	-2.24	-2.55	-2.69
Number of Runs	49	46	42

a. Sequential Unconstrained Minimization Techniques (SUMT)

SUMT involves the creation of a pseudo-objective function that is the sum of the original objective function and a penalty function. This penalty function may be one of four types:

1) Exterior Penalty
2) Linear Extended Interior
3) Quadratic Extended Interior
4) Cubic Extended Interior

All these methods cause the objective function to have its actual value when the design variable is within the feasible region, but when the design variable is outside the feasible region, the penalty function significantly increases the objective function value. The penalty functions listed above vary in the value that will be added to the objective and how the transition between the feasible and infeasible regions is conducted [Ref. 3:p. 136].

b. Augmented Lagrange Multiplier (ALM)

In this method a pseudo-objective function is generated based upon the exterior penalty function, but with an additional term in order to create the

Lagrangian. It can be shown that the solution to the Lagrangian is the solution of the original optimization problem [Ref. 3:p. 141].

c. Sequential Linear Programming (SLP)

The basis of this strategy is the simple linearization of the objective function and constraints. Rectangular move limits are established and then linear programming techniques are applied to optimize the objective function [Ref. 3:p. 155].

d. Method of Centers

The SLP method can produce infeasible designs with each iteration. To correct this problem the Method of Centers generates an inscribed hypersphere based upon the linearized objective function and constraints, then uses linear programming techniques to determine the optimum.

e. Sequential Programming

Sequential Programming can be of either of two types:

1) Sequential Quadratic
2) Sequential Convex

Both of these methods refine the method for determining the search direction for the objective function minimum. In the quadratic method the objective function is transformed into a quadratic function, while the convex method generates the reciprocal of the objective function. These methods create conservative approximations to the actual optimization problem which can then be easily evaluated [Ref. 4:p. 2].

2. **Optimizers**

The method by which new design variables are chosen is a result of the optimizer selected. Each of the possible methods predicts a new set of design variables which should be closer to the optimum design, based upon evaluations of the objective function at previous values of the design variables.

a. Conjugate Direction Method

This optimizer, developed by Fletcher and Reeves [Ref. 6:pp. 149-154], is based upon finding the negative gradient of the objective function from some initial point and then performing a one-dimensional search in that direction for the minimum value of the objective function. This process is repeated from the newly found point, and then a new direction is determined based on the previous two points and directions.

TABLE 2

ADS OPTIONS

Strategy (ISTRAT)

0 - None

1 - SUMT, Exterior Penalty Function

2 - SUMT, Linear Extended Interior

3 - SUMT, Quadratic Extended Interior

4 - Cubic Extended Interior

5 - Augmented Lagrange Multiplier Method

6 - Sequential Linear Programming

7 - Method of Centers

8 - Sequential Quadratic Programming

9 - Sequential Convex Programming

Optimizer (IOPT)

1 - Fletcher-Reeves

2 - Davidon-Fletcher-Powell (DFP)

3 - Broydon-Fletcher-Golfarb-Shanno (BFGS)

4 - Method of Feasible Directions

5 - Modified Method of Feasible Directions

One dimensional Search (IONED)

1 - Golden Section Method

2 - Golden Section and Polynomial

3 - Polynomial Interpolation, bounded

4 - Polynomial Extrapolation

5 - Golden Section Method

6 - Golden Section and Polynomial

7 - Polynomial Interpolation, bounded

8 - Polynomial Extrapolation

Note: One dimensional searches 1 through 4 are for use with Optimizers 1 through 3, and searches 5 through 8 are for use with Optimizers 4 and 5.

Description Manual [Ref. 4: pp.3-5]. In ADS, each possible combination is identified by use of a three digit code (XYZ) with the X digit representing the optimization strategy, the Y digit the optimizer, and the Z digit the one-dimensional search employed.

Vanderplaats [Ref. 4] points out that although there is a multitude of different combinations, eight combinations have, as a result of experience, become recommended for general applications. These eight combinations were tested using otherwise identical ADSL programs. The results are presented in Table 3.

TABLE 3
CASE 1 ADS STRATEGY RESULTS

Strategy	047	057	857	957	533	233	133	656
Objective	1.966	1.966	1.985	1.959	1.964	1.964	1.964	1.972
Design Variable	0.720	0.719	0.703	0.729	0.722	0.722	0.722	0.714
Constraint 1	-.362	-.361	-.355	-.365	-.362	-.362	-.362	-.359
Constraint 2	-.157	-.157	-.150	-.054	-.158	-.158	-.158	-.154
Constraint 3	-.077	-.089	-.207	-.008	-.057	-.069	-.069	-.126
Constraint 4	-.009	0.000	-.036	+.048	-.018	-.014	-.014	+.033
Constraint 5	-2.69	-2.84	-2.95	-2.49	-2.64	-2.61	-2.61	-2.89
Number of Runs	10	27	11	10	47	51	51	29

Observations that can be made are:

- All the combinations found essentially the same optimum value.
- There are significant differences in the efficiency of the various methods. If the criteria of minimum number of objective function evaluations is used, the 047 and 957 strategies are the best.
- All the combinations found solutions that met all the constraints within the ADS default tolerance band.
- More objective function evaluations did not necessarily lead to reduction in objective function value. In fact, one of the quickest methods (957), found the smallest value.

F. PERFORMANCE INDEX SELECTION

Ogata [Ref. 5: pp. 296-301]. presents four performance indices that are based upon the time-integrated weighted error between the desired output and the actual system response. These indices are:

- Integral square-error (ISE)
- Integral-of-time-multiplied square-error (ITSE)
- Integral absolute-error (IAE)
- Integral-of-time-multiplied absolute-error (ITAE)

These performance indices vary in complexity for analytical evaluation over the integration period of zero to infinity. For two of the cases, namely ISE and ITSE, exact analytical answers can be found as indicated in Table 4. By using ADSL the analytic problems are greatly simplified for the other indices.

TABLE 4
CASE 1 PERFORMANCE INDEX RESULTS

Index	*IAE*	*ISE*	*ITAE*	*ITSE*
Strategy	047	047	047	047
ADSL Objective	1.605	1.014	1.966	.707
Exact Objective	n.a.	1.000	n.a.	.707
Design Variable (ζ)	.6605	.5909	.7202	.5928
Exact ζ	n.a.	.5000	n.a.	.5946
Constraint 1	-.337	-.300	-.362	-.301
Constraint 2	-.227	-.152	-.157	-.207
Constraint 3	-.447	-.858	-.077	-.831
Constraint 4	-.299	-.717	-.009	-.646
Constraint 5	-3.32	-3.64	-2.69	-3.94
Number of Runs	23	40	10	45

The four performance indices were placed into identical ADSL programs as the objective function for optimization, and the results are shown in Table 4. Significant observations that can be drawn from this table are:

- As expected, each of the methods found a different optimum value for the design variable.
- All of the methods found an optimum value that did not violate any constraint.

- Within numerical tolerance, the results are the same as those found by Ogata [Ref. 5: p. 300].
- The apparent discrepancy in optimum damping ratio for the ISE performance index is due to the flat minimum of the objective function in this case (a difference of 20% in ζ gives a performance index variation of less than 2%).

It should be noted that the magnitude of the resulting objective functions cannot be compared directly to determine which method is "better." This would be like comparing apples and oranges, due to the significant mathematical difference in the objective functions/performance indices.

VI. INTEGRAL FEEDBACK CONTROL DESIGN

A. STATE SPACE ANALYSIS

The time and frequency domain methods discussed in the previous chapter are most useful in the analysis of single input, single output (SISO) systems. State-space methods give added flexibility in controller design and allow the treatment of complex, multiple input, multiple output (MIMO) systems. The concept of state, which is the basis of modern control theory, embodies the following key elements:

- State Variables: the smallest set of n variables, with their initial quantities known, that completely describes the behavior of a dynamic system for a given input over time. It should be noted that these states do not necessarily have to be physically accessible.
- State: the condition of a dynamic system at a given time (t) that is uniquely determined by the initial state of the system and the inputs to the system after the initial time (t_0).
- State Vector: the vector made up of the n state variables needed to describe the system. The state vector is usually designated **x(t)**.
- Input Vector: the vector made up of the m variables that are inputs to the system. The input vector is usually designated **u(t)**.
- State Space: is the n dimensional space made up of the coordinate axes corresponding the individual states.

Kuo [Ref. 7] describes two types of basic systems that are typical of modern control systems:

1) State feedback control
2) State feedback with integral control

These systems have been modeled using the combined Dynamic Simulation Language (DSL) and Automated Design Synthesis (ADS) optimization code ADSL.

B. CASE 2 STATE FEEDBACK WITH INTEGRAL CONTROL

Case 2 is the next problem to be investigated. In this system control is attained by feedback of the state variables through constant gains and an integrator. A signal flow graph representative of state feedback with integral control is shown in Fig. 6.1. As a matter of interest, if the integrator is removed the resulting system is called a state

feedback controller. When the reference for any state feedback controller is set to zero, the resulting system is classified as a regulator. It can further be shown that the common proportional-integral-derivative (PID) and rate-feedback control systems are special cases of state feedback control [Ref. 7:p. 522].

Case 2 systems are of interest because the state feedback systems are generally only useful as regulators that have no noise input. Integral control suppresses the fluctuations caused by undesirable noise and thus provides for a much more effective system.

1. Model Description

The system selected for optimization was a dc electric motor speed control system proposed by Kuo [Ref. 7:pp. 532-536]. In this example the motor shaft angular velocity and armature current were selected as the state variables x_1 and x_2 respectively. Two inputs were considered:

- w_1 as a direct, torque loading acting on the motor shaft
- w_2 as the desired constant speed set point value.

The control and output variables can respectively be represented by the following relationships where g_1, g_2, and g_3 are the feedback gains, and the state variables are x_1 and x_2:

- $u(t) = -g_1x_1 - g_2x_2 - g_3\int c(t)dt$
- $c(t) = w_2 - x_1$.

The corresponding signal flow graph showing the various relations in the control system is shown in Fig. 6.1. The initial conditions and values assumed for the design are shown in Tables 5 through 8. The resulting state responses are displayed in Fig. 6.2.

The goal of the optimization is to determine the values of the three gains -- g_1, g_2, and g_3 -- necessary to minimize a stated objective function without violating specified constraints for controller performance and stability.

2. ADSL Program

Appendix D contains the source code listing of the ADSL program which models the state feedback system with integral control for Case 2. It follows the same form of construction as used for Case 1. The initialization of parameters was expanded into three distinct groupings:

1) Array dimensioning and parameter initialization for ADS
2) DSL control parameter setting

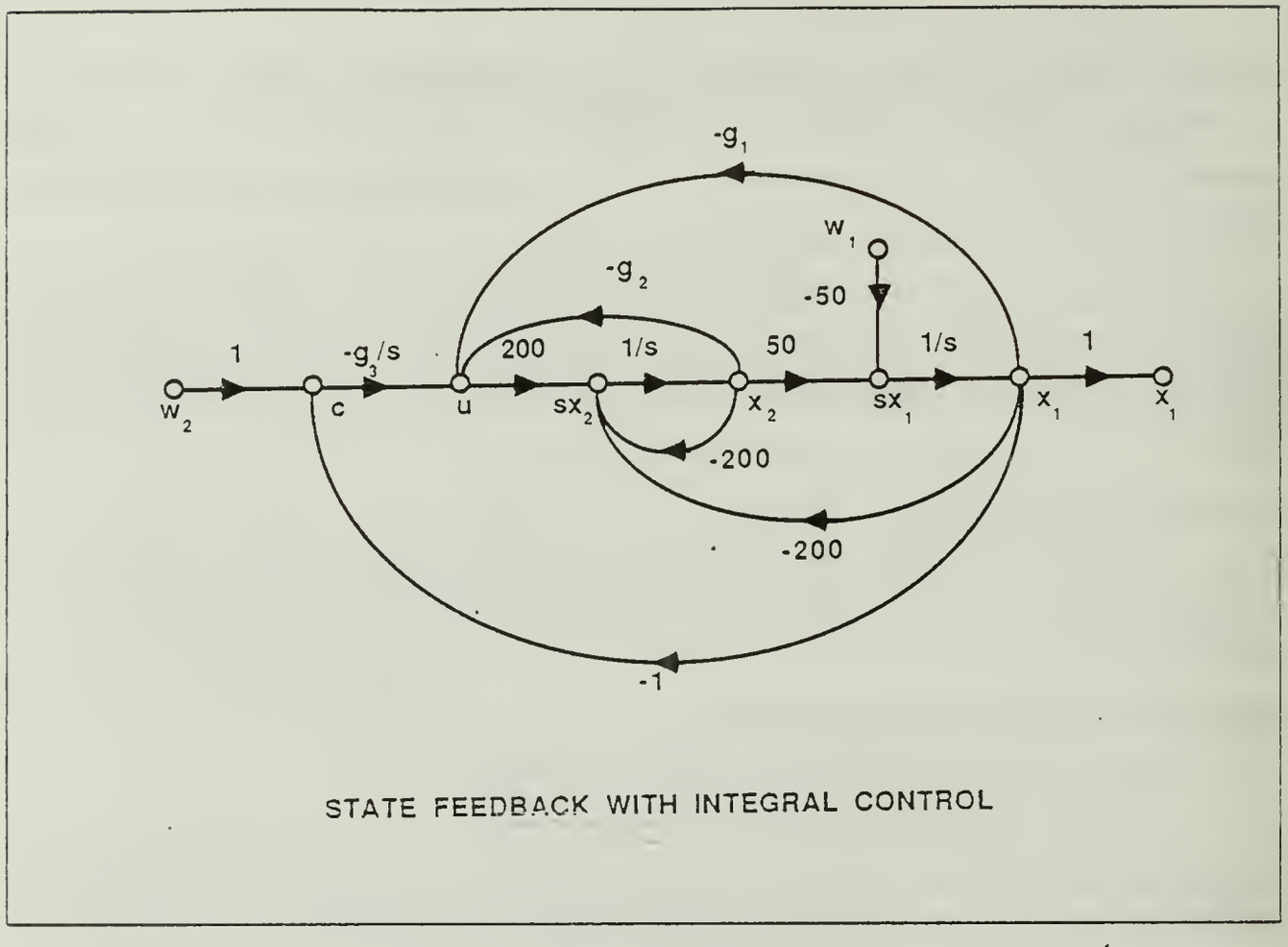


Figure 6.1 Case 2 Signal Flow Graph

3) Model description, constraint evaluation, and integration completion control parameter implementation.

Using the results from Case 1, selection of the stiff integration method was made. An additional change was to reduce both the relative and absolute error criteria for the integration routine from 1.0E-05 to 1.0E-04. This was done in order to increase computational efficiency while still providing for an order of magnitude greater precision than that expected for the objective function optimization done by the ADS program, using the default parameter settings.

3. **Performance Index/Objective Function**

For optimizing, the classical quadratic performance index (QPI) was adopted. This performance index: $J = {}_0\int^{\infty}(x^tQx + u^tRu)dt$ allows the user to specify individual weights for the Q and R matrices. Thus the desired relative importance between the error seen in each of the actual state values to the desired state values, and for the energy use associated with the control variable over time can be specified. For

TABLE 5
CASE 2 INITIAL CONDITIONS

Motor Shaft Angular Velocity (x_1) = 0.0 rad/s

Armature Current (x_2) = 0.0 Amps

Feedback Control Signal (u(t)) = 0.0 Volts

Input Disturbance Torque (w_1) = 0.0 N-m

Input Reference Set Point Speed (w_2) = 1.0 rad/s

TABLE 6
CASE 2 INITIAL POLE SELECTION

Initial Pole Selection

-10.0+ 10.0j -10.0-10.0j -300.0

Exact Resulting Initial Gains

g_1 = -0.38 g_2 = 0.60 g_3 = -6.00

Case 2 each state was selected to be independent of the other, and thus Q and R were identity matrices. Table 8 shows each of the matrices used in the Case 2 performance index.

4. **Exact Initial Gain Determination**

ADS requires that an initial guess for the design variables be provided in order to start the optimization. In conjunction with the the determination of these values a

TABLE 7
CASE 2 CONSTRAINTS

CSP = Maximum Value of the Output c(t) = 140% of initial value
TSP = Maximum Completion Time = 1.5 sec
USP = Maximum Value of the Control u(t) = 10.0 V

TABLE 8
CASE 2 PERFORMANCE INDEX MATRICES

$$Q = \begin{matrix} 1.0 & 0.0 \\ 0.0 & 1.0 \end{matrix} \qquad R = 1.0$$

$$x = \begin{matrix} w_2 - x_1 \\ x_2 \end{matrix} \qquad u = -g_1 x_1 - g_2 x_2 - g_3 \int c(t)dt$$

reasonable assumption has been made that the controller designer be able to specify stable poles for the system. Kuo [Ref. 7:pp. 527-528] outlines an analytic, general method for the determination of initial state feedback gains without transformation to the phase variable canonical form of the system characteristic equation. The method requires defining the following terms:

- Open loop characteristic equation:

  $\Delta_o = \text{Det}(s\,\mathbf{I} - \mathbf{A})$
- Closed loop characteristic equation with state feedback:

  $\Delta_c = \text{Det}(s\,\mathbf{I} - \mathbf{A} + \mathbf{BG})$
- Relation Matrix:

  $k(s) = [\,\text{Adj}\,(s\,\mathbf{I} - \mathbf{A})]\,\mathbf{B}$

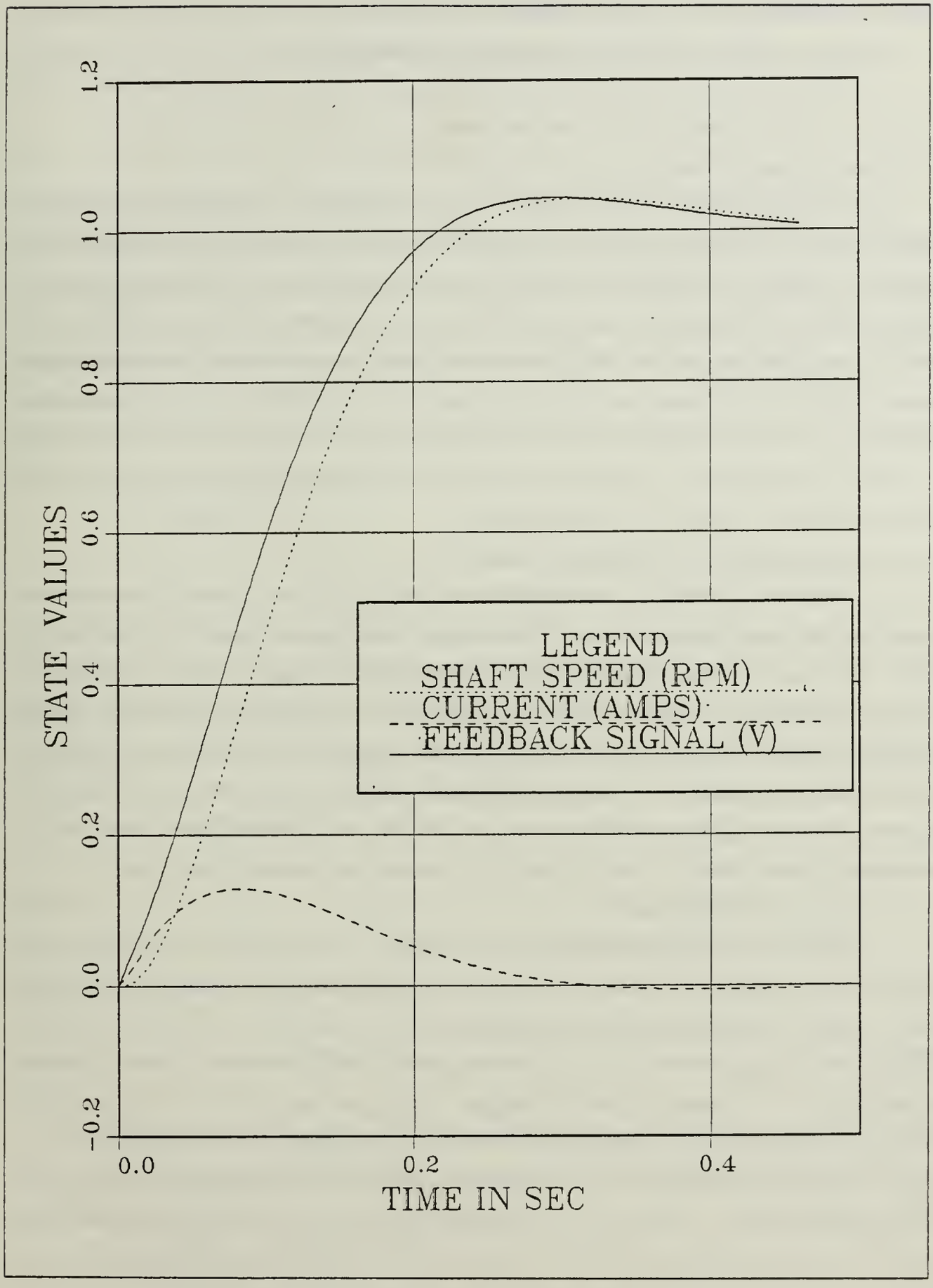


Figure 6.2 Case 2 Initial Condition State Responses

Where the indicated matrices are as follows for n states:

A- the coefficient matrix for the state variables (n x n).

B- the coefficient matrix for the control input (n x 1).

G- the state feedback gain matrix (1 x n).

I- the identity matrix.

The relationship $Gk(s) = \Delta_c(s) - \Delta_o(s)$ can then be solved for G using the specified initial design system poles as the eigenvalues for the closed loop characteristic equation. In the Case 2 ADSL program, shown in Appendix D, the necessary matrix operations have been multiplied-out and the like terms relating the various powers of *s* have been grouped together to yield a system of three equations and three unknowns. Matrix solver routines from the LINPAC library [Ref. 8], DGEFA (double precision general matrix conditioning) and DGESL (double precision general matrix solver), were then called to respectively decompose and solve the system of equations, and thus generate the exact initial state feedback gains.

5. Approximate Initial Gain Determination

The matrix method just described above works well, and is an efficient method for systems of order three or less. With higher order systems the determination of adjoint matrices is difficult and time consuming. To counter this problem, an alternative method was devised. The determination of the gains was considered as an optimization problem, where the constraints and objective function were formulated such that the there would be a minimum difference between the poles corresponding to the gains determined by optimization using ADS, and the original poles specified by the designer. Two different forms of the optimization problem were developed.

a. Determinant Method

For the first method, the complex matrix: (*s* I - A + BG) was formed by substituting in one of the desired poles for *s* and setting the initial values for the gains to 1.0. The determinant of this matrix was then found using the LINPACK routines ZGEFA (complex general matrix conditioning) and ZGEDI (complex general matrix derterminant). If the correct value of the gain, corresponding to the desired pole was substituted into the matrix equation, the resulting determinant would be equal to zero. Thus an equality constraint can be established equal to the absolute value of the resulting determinant. This process was then repeated for each of the other desired poles. The objective function was then defined as the sum of all the constraint values. For an exact solution the objective function would be equal to the value of zero. ADS

was then called to vary the gain values to find the minimum value of the objective function. The final determined gains thus should represent an approximate match to the desired initial poles.

When this algorithm was attempted, the results were not as expected. Table 9 shows the results for various initial poles using the Case 2 model. Looking at possible reasons for these outcomes it can be found that the optimization problem, as formulated above, was very ill conditioned. When the exact answers were substituted as initial guess for the gains, the determinant values for each gain resulted a machine zero of 1.0E-08. A very small deviation of 0.01 from the exact gains was then tried and resulted in a determinant value of greater than 1.0E+06, as did the determinant for the initial guess gain values of 1.0. These radically different determinant values produced very large gradients for ADS to work with, and in turn, ADS could not find a solution that closely approximated the desired initial poles.

b. Eigenvalue Method

In order to overcome the ill conditioning problem, a more robust method was devised. The real matrix: (A - BG) was formed, using initial guess values for the gains of -1.0. The eigenvalues of this matrix are the corresponding poles of the system, and were then found using the Eigensystem Routines (EISPACK) subroutine RG (real general matrix solver) [Ref. 9]. The resulting largest and smallest, complex eigenvalues were then compared to the largest and smallest, desired complex poles using the FORTRAN min and max functions for both the real and imaginary parts. Four equality constraints were then defined as the absolute differences between the corresponding min or max, and real or imaginary parts of the desired poles and calculated eigenvalues. The objective function was then set equal to the sum of the constraint values. ADS was then called to minimize the objective, and thus find the gains corresponding to the desired poles. Table 10 shows the results for this method using the same initial conditions as in the determinant method above. It can be seen that this method finds reasonably accurate values for the gains, compared to the exact method.

It should be pointed out that in order to prevent the the optimizer from finding a local minimum corresponding to a positive real pole solution the following mechanisms were programmed into the algorithm:

- Strategy 133 was selected.

TABLE 9

CASE 2 INITIAL GAIN APPROXIMATION USING DETERMINANTS

Desired Pole	*Derived Pole*	*Derived Gains*	*Exact Gains*
-10.0 + 10.0j	-14.33 + 9.66j	-.125	-.380
-10.0-10.0j	-14.33-9.66j	+ .617	+ .600
-300.0	-294.8	-8.81	-6.00
-10.0 + 10.0j	+ .257	+ .414	-.920
-10.0-10.0j	-473.2	+ 1.51	-.750
-30.0	-30.1	+ .367	-.600
-10.0 + 10.0j	-24.15	+ .391	+ 5.02
-10.0-10.0j	-582.9	+ 2.03	+ 14.1
-3000.0	+ .270	+ .360	-60.0
-100.0 + 10.0j	-99.00	+ .167	+ 1.50
-100.0-10.0j	-274.7	+ .863	+ 6.01
-300.0	+ 1.08	+ 2.94	-303.0
-10.0 + 100.0j	-66.2	+ .362	+ .610
-10.0-100.0j	-10.59	+ .343	+ .600
-300.0	-201.4	-14.13	-303.0

ADS Strategy 957

- The default ADS scaling of the optimization problem was disabled (to be discussed in a latter section).
- A check for positive real parts of the smallest eigenvalue was done. If a positive eigenvalue was found, then the sign of the first gain value was reversed.

An observation that should be pointed out, as a consequence of these determinations, is how sensitive optimization is to problem definition. Mathematically, both of the methods discussed above are equally good for establishing the gains corresponding to the desired poles, but the second method results in always dealing with numerical values of essentially the same order of magnitude, and a much better conditioned problem for ADS to solve. The source code for both of the approximate gain determination methods has been included in Appendix E.

TABLE 10

INITIAL GAIN DETERMINATION USING EIGENVALUES

Desired Pole	*Derived Pole*	*Derived Gains*	*Exact Gains*
-10.0 + 10.0j	-9.99 + 10.05j	-.384	-.380
-10.0-10.0j	-9.99-10.05j	+ .599	+ .600
-300.0	-300.0	-5.98	-6.00
-10.0 + 10.0j	-7.84 + 9.48j	-.939	-.920
-10.0-10.0j	-7.84-9.48j	-.775	-.750
-30.0	-29.2	-.442	-.600
-10.0 + 10.0j	-118.1	+ 37.6	+ 5.02
-10.0-10.0j	-10.3	+ 14.6	+ 14.1
-3000.0	-3000.0	-364.0	-60.0
-100.0 + 10.0j	-65.3 + 8.76j	+ 1.11	+ 1.50
-100.0-10.0j	-65.3-8.76j	+ 3.25	+ 6.01
-300.0	-292.0	-127.1	-303.0
-10.0 + 100.0j	-10.0 + 99.7j	+ .603	+ .610
-10.0-100.0j	-10.0-99.7j	+ .598	+ .600
-300.0	-299.9	-301.0	-303.0

ADS Strategy 133

6. **Constraint Definitions**

The maximum overshoot and completion time constraints, as previously described, were applied directly to this problem. Using a manner similar to the maximum overshoot, a new constraint was defined for the maximum allowable value of the feedback control signal u(t).

An additional feature seen in the state feedback problem is that as ADS begins its optimization it has no way of knowing whether or not the design variable values proposed for the feedback gains will cause the system to become unstable. To account for this, two options are available:

1) In a reverse manner to the initial gain computations the ADS proposed gains could be used to compute the new corresponding system poles. These poles could then be evaluated for stability.

2) An extreme upper limit could be set for the allowable system output in order to monitor for stability. When this condition was reached the run could be terminated and an unstable solution equality constraint could be invoked.

This second method was selected as more numerically efficient as it can be assumed that most proposed gains would be reasonably stable, and the cost of pole evaluation for every set of ADS proposed gains is high due to the need for finding eigenvalues to the closed loop characteristic equation.

A stability constraint was proposed, in private discussions with Vanderplaats, as a way to eliminate unstable solutions. A simple way to look at this method is to consider the total design space. The constraints related to specific performance (such as completion time) define the borders of the space, and the resulting interior region is the domain of feasible solutions. Addition of the stability constraint essentially generates a "hole" in the feasible region by eliminating possible solutions at specific points, while allowing solutions at points close by. Normally the stability constraint would be set to a constant, negative value of say minus one, and thus it would never define an active boundary of the design space. However, when an invalid design is proposed by ADS, through a request for evaluation of controller gains that result in violating the maximum allowable output (an unstable condition), the stability constraint would be set to a positive one, and the previous value of the objective function would be maintained. When ADS sees a step change in the unstable solution constraint, it will adjust the design variables to return this constraint negative. The result will be that only valid values of the design variables are considered for evaluation.

Nye and Tits [Ref. 10:p. 1695] proposed the definition of hard and soft constraints. All the previously defined constraints for system performance (maximum overshoot, completion time, etc.) fall into the category of soft constraints, because they are set by the designer to meet flexible goals through the optimization. One type of hard constraint they consider is that of negative response of the input. This constraint simply checks to see that the response curve takes off in the same direction that the input goes in, otherwise the solution will be divergent. This constraint was implemented in a similar manner to the stability constraint check above.

7. Integration Termination

With the introduction of integral feedback and multiple inputs, the performance index/objective function can become a ramp, rather than having a plateau

as in Case 1. To account for this effect, an additional check for a constant second derivative of the objective function was added to the model as a possible criterion for the simulation integration termination.

8. Final Pole Determinations

Once ADS completes the optimization, the final system gains are known. In order to give the designer information on how much the optimization caused the systems poles to be changed from the originally specified ones, the program calculates the final system poles. This requires a two step process:

1) Calculation of the the final closed loop system matrix equation: A - BG
2) Determination of the resulting matrix eigenvalues by the same procedure described in the approximate initial pole to gain determination using eigenvalues.

C. SCALING

In work by Chow [Ref. 11] and Gordon [Ref. 12], where ADS was used in the analysis of pole placement techniques, scaling was "turned off" by manipulation of the ADS integer work matrix (IWK). The technique to perform this operation is described in the ADS Manual [Ref. 4:pp. 12-16]. One reason for not scaling is to increase computational efficiency, as the scaling calculations are not required and thus directly reduce the time required to run the problem. It was felt that a determination should be made as to how scaling effects the optimization problem when using ADSL.

1. Scaling the General Optimization Problem

Scaling of any numerical problem is generally considered good engineering practice. Vanderplaats [Ref. 3:pp. 136-137] describes in detail the theory of scaling the optimization problem by two different methods. First, the initial problem can be scaled to provide relatively the same order of magnitude to the design variables, constraints, and objective function. In Case 2 this was done in the initial problem statement. The second scaling is then done to the gradients of the individual parameters, so that they also have the same order of magnitude. Scaling of the gradients, in addition to the values, is critical because the basic optimization problem rests on finding a minimum, or in other words, finding where the gradient goes to zero.

2. Scaling in ADS

Scaling within ADS is done by applying separate scaling factors to the objective function, and each design variable and constraint. From examination of the

ADS source code [Ref. 13], the design variable and constraint scaling factors were found to be applied as follows:

1) Gradients are found using finite differences based upon the initial ADS call values.
2) The absolute value of the gradient is taken.
3) If the gradient is less than the ADS numerical value for zero, the gradient value is set to 0.1.
4) If the gradient is greater than 1000.0, the gradient value is set to 1000.0.
5) The scale factor is set equal to the inverse of the gradient value.
6) The product of each design variable and its respective scale factor is taken.
7) The product of each constraint and its respective scale factor is taken.
8) All scale factors are maintained unchanged throughout the optimization.
9) Upon completion of optimization, each design variable and constraint is divided by its respective scale factor, to provide a final value that is not scaled.

For the objective function the scale factor is based upon the square root of the sum of the squares of the design variable scale factors. When scaling is not used, ADS sets the scale factors to a value of 1.0.

3. **Scaling in Case 2**

In order to investigate how scaling would affect the problem posed in Case 2, the initial design noted in Tables 6 through 8 was assumed to provided a reasonably scaled problem. Various strategy combinations were then selected and optimized with and without scaling applied. Table 11 shows the results of these runs, and Fig. 6.3 is a plot of the state responses for the optimized controller, using initial poles of -10.0+ 10.0j, -10.0-10.0j, and -300.0 and the 047 strategy for optimization.

Trends that can be seen from these results are:

- In all cases the strategy that had scaling applied found a lower value of the objective function.
- All strategies found essentially the same optimum if there was no scaling.
- By scaling, and using the most costly strategy (133) with respect to total simulations required, the lowest objective function value was found.

In Appendix F the output listing for the 057 strategy combination with and without scaling can be found. Close inspection of the iteration to iteration optimization results reveals that the first iteration is the most critical to the problem solution. In the run without scaling, the optimization proceeds slowly, and relatively

TABLE 11

CASE 2 SCALING RESULTS

Strategy	*Scaling*	*Objective*	*Final Poles*	*Total Simulations*
057	Yes	.0492	-20.6 + 66.2j -20.6-66.2j -31.93	69
057	No	.0658	-11.5 + 52.2j -11.5-52.2j -21.75	67
047	Yes	.0492	-20.6 + 66.3j -20.6-66.3j -31.93	69
047	No	.0640	-11.1 + 52.8j -11.1-52.8j -21.77	79
957	Yes	.0432	-22.5 + 75.5j -22.5-75.5j -42.57	76
957	No	.0543	-15.4 + 61.9j -15.4-61.9j -28.7	80
133	Yes	.0250	-346. + 383.j -346.-383.j -366.2	136
133	No	.0645	-9.35 + 54.4j -3.35-54.4j -20.9	101

Initial Poles

-10.0 + 10.0j -10.0-10.0j -300.0

Initial Objective = 0.1396

constantly towards the minimum. The gradients are small and thus provide less significant data for ADS to process in the determination of search direction and step size. When scaling is applied, a significant initial step is made. In fact, the initial step "jumps over" the optimum found in the case without scaling, and then simple refinement is done. Thus the classical case of a relative minimum being found is seen.

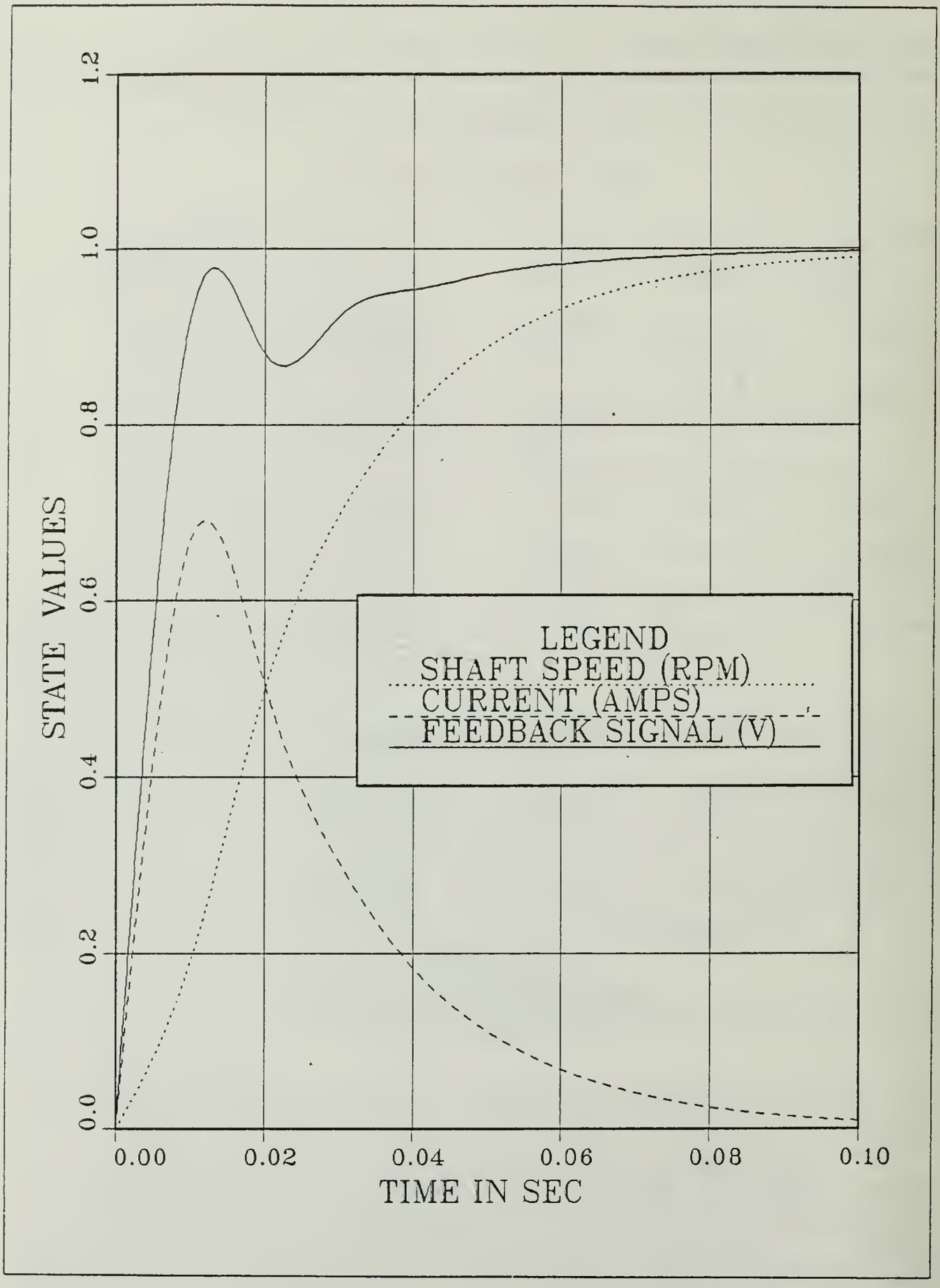


Figure 6.3 Case 2 Optimal Response

It can be concluded that the computational efficiency of turning scaling off is not beneficial when using ADSL.

A word of precaution about the use of ADS print control parameter (IPRINT) [Ref. 4:p. 10] is in order. The value assigned to this parameter should be selected to provide a printout that reveals, at least, the optimizer termination criteria. Use of IPRINT = 1000, for example, prints only the initial and final optimization results. The maximum iteration termination could then be invoked and the user would not know that the result obtained from ADS was questionable. While IPRINT = 3050 produces more output than is really useful, it is valuable for checking what is really happening during the optimization process, the type of scaling that occurs, and to reveal the complete status of the solution. In general, use of IPRINT = 1010 for cases when no strategy is used (ISTRAT = 047 or 057) or IPRINT = 1100 for cases where a strategy is used (ISTRAT = 133 or 957), proved out to be the most beneficial print control values.

D. INITIAL POLE SELECTION

Different initial poles were selected to determine how initial selection effects the optimization process. Intuition may lead one to assume that the optimizer should find the same result no matter where the initial poles are placed. This estimate turns out to be incorrect on several counts. First, the optimizer is likely to find the minimum value of the objective function closest to where it initially starts. This is know as a relative or local minimum. Without further investigating the whole spectrum of possible values for the design variables, the optimizer has no way of differentiating between a relative minimum, and the global minimum, which represents the minimum of all relative minimums of the objective function [Ref. 3:pp. 12-13]. In Case 2, changing the initial poles results in corresponding gains being sent to ADS that can vary drastically. A significant change in the initial design then leads to solving essentially a completely different optimization problem, that could have its own distinct relative minimum. Whether or not this minimum is the global minimum is not known. Secondly, the poles selected may result in an initial solution in which one or more constraints are not met. This is know as starting in the infeasible domain. ADS will attempt to find a feasible solution that will meet all the constraints, but in some cases it is unsuccessful. In the cases where ADS cannot meet all the constraints, it will still attempt to find the best solution, but with one or more violations present. This means that although ADS

found a solution, the designer should still try a different set of initial poles in an attempt to start ADS in the feasible domain. A shift in the initial pole placement can generate a solution that is significantly better, and one that meets all constraints. Table 12, shows the results for both small and large changes in the initial guess for the system poles using the Case 2 model. Figures 6.4a and 6.4b shows the pole placement results graphically (not to scale).

TABLE 12

CASE 2 POLE SELECTION RESULTS

Initial Poles	*Initial Gains*	*Initial Objective*	*Final Poles*	*Final Gains*	*Final Objective*
-10.0+10.0j -10.0-10.0j -300.0	-.380 +.600 -6.00	.139	-344.2+342.0j -344.2-342.0j -40.88	+25.3 +2.64 -962.4	.024
-10.0+10.0j -10.0-10.0j -30.0	-.920 -.750 -.600	.189	-3.17+25.9j -3.17-25.9j -10.2	-.925 -.917 -.699	.143
-10.0+10.0j -10.0-10.0j -3000.0	+5.02 +14.1 -60.0	.133	-75.2+150.8 -75.2-150.8j -46.7	+2.54 -.013 -132.8	.029
-100.0+10.0j -100.0-10.0j -300.0	+1.50 +6.01 -303.0	.026	-386.5+165.3j -386.5-165.3j -60.4	+20.9 +3.14 -642.4	.023
-10.0+100.0j -10.0-100.0j -300.0	+.610 +.600 -303.0	.181	-248.8+210.7j -248.8-210.7j -60.4	+12.6 +1.79 -624.4	.024

ADS Strategy 047

Analysis of the results shows that:

- The assumed initial poles of: -10.0+10.0j, -10.0-10.0j, and -300.0 provided a fairly reasonable value for the optimized objective function.
- If the complex poles are dominant, nearly the same value for the objective function was found for every case of selecting different initial poles.

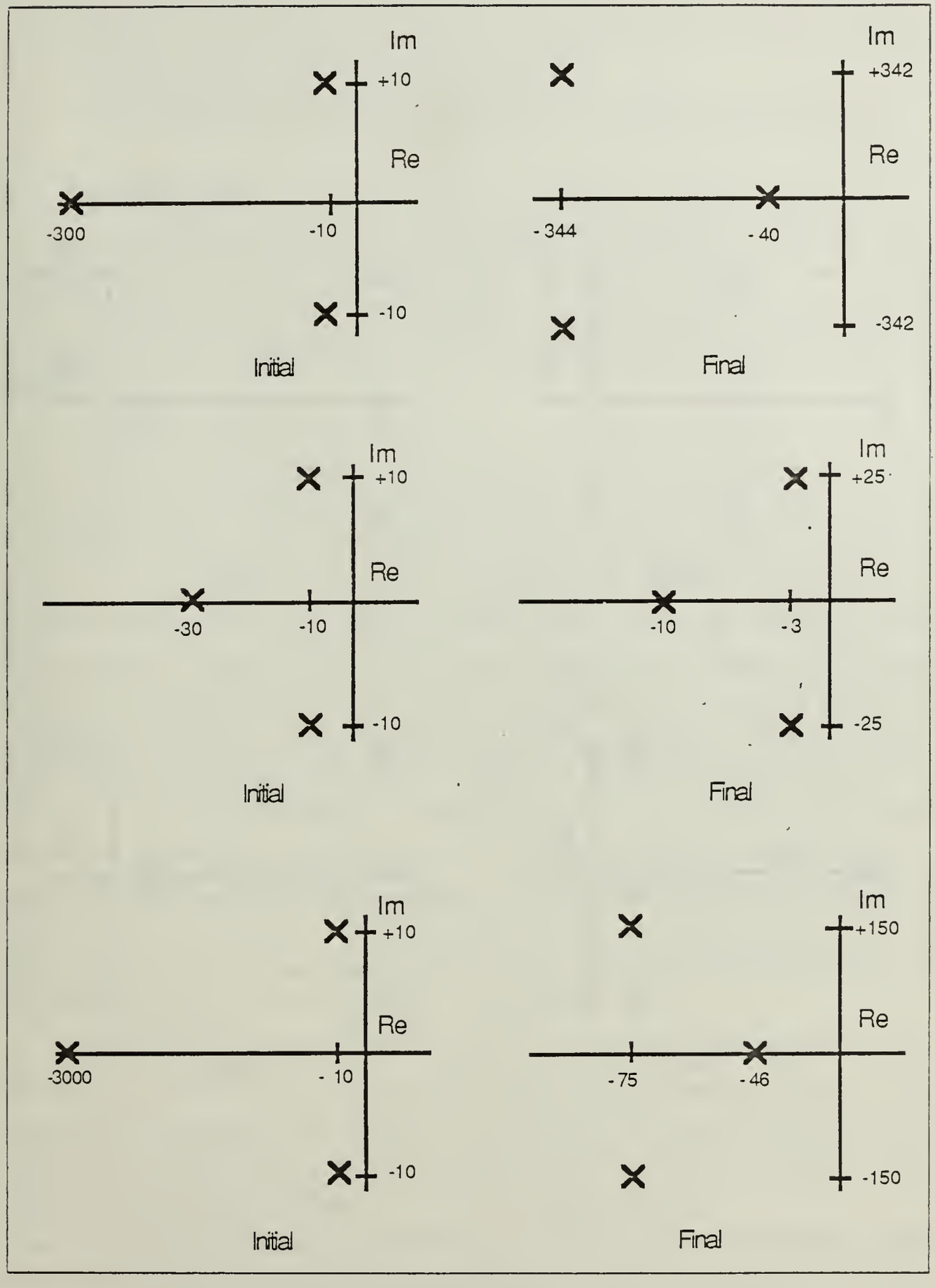


Figure 6.4a Case 2 Initial Pole Selection Results

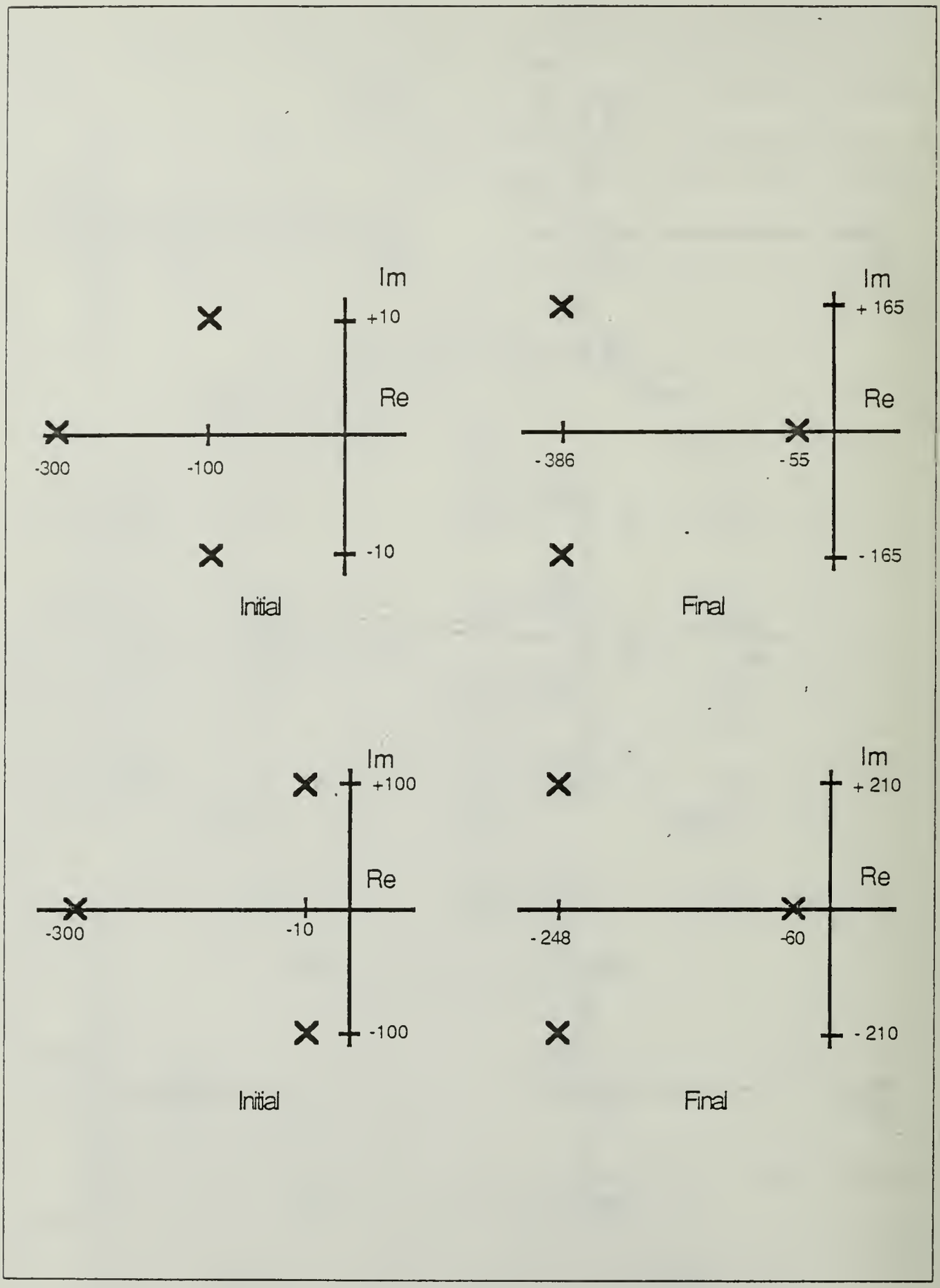


Figure 6.4b Case 2 Initial Pole Selection Results (cont.)

- There was a wide variety of initial pole combinations that resulted in essentially the same value for the final objective function.
- The average values for the resulting poles are -50, -250 + 150j, and -250-150j.

E. CONSTRAINTS

ADS considers two general types of constraint violations: The first, side constraints, are based upon the upper and lower bounds applied to each design variable (VUB and VLB). The second, defined constraints, are those relationships of the design variables to specific limits, that are developed by the designer to specify the system.

1. Side Constraints

Through side constraints, ADS prevents the optimization process from using a value less than the specified lower bound or greater than the desired upper bound for each design variable. The values of these bonds are specified based upon physical restrictions to the system. For Case 2, these restrictions would be derived from the physical circuitry used to provide the feedback gains. If at the beginning of the optimization, the initial value of any design variable is set outside the allowable bounds, the design variable is reset to the closest bound, and a warning message is generated. It should be noted that this situation can easily occur using the Case 2 ADSL model. The designer will specify a stable pole combination, and in general will not have a good idea of what the corresponding gain values are. In reality the gain values will be physically limited to finite values of usually only several orders of magnitude. To explore how the optimization is affected by restrictive side constraints multiple runs of the Case 2 model, described by Tables 5 through 8, were conducted with varying upper and lower bound values. Table 13 shows the results for order of magnitude changes in both the upper and lower bounds.

Examination of these results shows that by making the side constraints more restrictive, a direct effect can be seen on the objective function value. As the constraints are tightened, the objective function grows in value, indicating that the system has a greater amount of error associated with it over time, and that a larger feedback control signal is required to achieve the desired response.

As a word of caution, it should be noted that in order to suppress any design variables from the model, the programmer must directly reduce the number of design variables (NDV). Simply putting the upper and lower bounds (VUB and VLB) to the

TABLE 13

CASE 2 RESULTS FOR REDUCING DESIGN VARIABLE BOUNDS

Upper Bound	*Lower Bound*	*Active Constraint*	*Final Objective*	*Final Poles*	*Final Gains*
+ 1000.0	-1000.0	none	0.0233	-381 + 247j -381-247j -47.2	23.3 3.05 -978.1
+ 100.0	-100.0	g_3	0.0305	-61.9 + 135j -61.9-135j -44.9	1.78 -0.156 -100.0
+ 10.0	-10.0	g_3	0.0561	-12.2 + 62.2j -12.2-62.2j -24.8	-0.536 -0.752 -10.0

Initial Poles

-10.0 + 10.0j -10.0-10.0j -300.0

ADS Strategy 047

same value may seem to eliminate the variable, but it will still be taken into account. ADS will calculate the corresponding gradients and include them in the optimization with unpredictable results.

2. **Defined Constraints**

The next area to be considered was how does reducing individual constraints influence the optimization process. It should first be pointed out that ADS keeps tracks of constraints in two ways:

1) Violated constraints: a constraint is considered as violated when its value exceeds 0.0 + CTMIN, where CTMIN is defaulted to a value of 0.01 for non-linear constraints, and 0.001 for linear constraints.

2) Active constraints: a constraint is considered as active when its value exceeds 0.0 + CT, where CT is defaulted to -0.03 for non-linear constraints, and -0.005 for linear constraints.

CTMIN and CT may be modified by changing their corresponding WK array values [Ref. 4:pp. 12-14].

Recalling that all equality constraints, whether linear or non-linear, must be equal to zero, and all inequality constraints must be equal to, or less than zero, the CT and CTMIN values can then be considered to generate a tolerance band for ADS to work within. Thus ADS never exactly meets an equality constraint of zero, but rather finds a value in the band between CT and CTMIN. In the the CONMIN program this band was known as the "constraint thickness."

Constraint thickness comes into play for the Case 2 model with the specifications for maximum response (CSP), completion time (TSP), and maximum feedback control (USP). If these specifications are made too small, they will essentially become lost within the constraint thickness boundaries. In order to look at this effect with Case 2 an additional constraint was added after noting the findings about the x_2 (motor current) state in the initial pole investigation. In a manner similar to that used for the maximum feedback control signal constraint, a maximum value for state x_2 (X2SP) was established. This constraint was then tightened in order to observe its effect on the objective function and the other constraints. For this investigation, the CSP, TSP, and USP values were reduced to where they resulted in almost active constraints for the initial conditions and design described by Tables 5 through 8. The exact values are listed in Table 14, and Table 15 documents the results of the simulations conducted for reduced values of X2SP.

TABLE 14

CASE 2 CONSTRAINT SPECIFICATIONS FOR X2SP REDUCTION

Constraint Specification	*Initial Model*	*Constraint Reduction Model*
CSP	140%	5%
TSP	1.5 sec	0.85 sec
USP	10 V	1.05 V

TABLE 15

CASE 2 X2SP REDUCTION RESULTS

X2SP	*Final Objective*	*Final Gains*	*Final Poles*	*Constraints*
0.75	0.0235	21.55 4.23 -1000.0	-772.0 -215.0 -60.2	USP active lower bound g_3 active
0.70	0.0235	27.99 4.17 -1000.0	-721.7 -259.9 -53.3	USP, CSP active lower bound g_3 active
0.60	0.0248	25.83 5.53 -996.0	-1089.0 -189.7 -48.2	X2SP active
0.40	0.0350	24.64 10.32 -681.1	-2146.0 -76.5 -41.4	X2SP active CSP active

Initial Poles

-10.0 + 10.0j -10.0-10.0j -300.0

ADS Strategy 047

From the results in Table 15, it appears that reducing the maximum allowable value for the x_2 state will have varying effects on the other constraints, but in general will cause the objective function to increase in value. This is as would be expected, because with a smaller allowable maximum motor current, the controller will see more error over time, and require a larger feedback signal to produce the desired output response. An additional note of interest is that for every case investigated, the final gains resulted in an optimum solution with all poles being real.

When the other constraints were tightened, individually, similar results were seen. It should be pointed out that once a constraint is pushed too far the model collapses. This collapse can be explained if one looks at the design variable space. With over restrictive constraints, the feasible design region disappears, and an optimum

design is impossible. An example of this phenomenon can be seen by looking at reducing the completion time constraint (TSP). Tables 16 and 17, respectively, show the initial conditions that were assumed and typical results.

TABLE 16

CASE 2 CONSTRAINT SPECIFICATIONS FOR TSP REDUCTION

Constraint Specification	*Initial Model*	*Constraint Reduction Model*
CSP	140%	5%
X2SP	n.a.	1.0 A
USP	10 V	1.05 V

TABLE 17

CASE 2 TSP REDUCTION RESULTS

TSP	*Final Objective*	*Final Gains*	*Final Poles*	*Constraints*
1.0	0.235	21.5 4.23 -1000.0	-772.0 -215.0 -60.2	USP active lower bound g_3 active
0.06	0.139	-.38 0.60 -6.0	-10+10j -10-10j -300.0	TSP violated

Initial Poles

-10.0+10.0j -10.0-10.0j -300.0

ADS Strategy 047

From the results of Table 17 it is apparent that if a constraint is tightened to the point where it becomes violated, the optimization will fail to find an improved solution. In most cases, ADS will stop after three iterations and return the initial solution as the final solution. This fact can be used as an aid when the design process is being conducted. The designer would see no improvement for the run and could then immediately use this result to relax the constraint specification and rerun the model. Typical process time for this cycle, when used with the Case 2 example, was one minute (including graphic display time for the IBM 3279 terminal).

F. INITIAL POLE SELECTION WITH ACTIVE CONSTRAINTS

In the previous discussion of initial pole selection, none of the proposed constraints were active due to the large values of respective specifications that were initially proposed. In order to compare the essentially unconstrained results of Table 12 to a more realistic problem, the Case 2 initial pole investigation was run again, but with the constraint specifications of Table 18 being applied. The results of these runs are shown in Table 19 and graphically in Figs. 6.5a and 6.5b (not to scale).

TABLE 18

CASE 2 CONSTRAINT SPECIFICATIONS FOR INITIAL POLE SELECTION

Constraint Specification	*Initial Model*	*Constraint Reduction Model*
CSP	140%	5%
X2SP	n.a.	0.6 A
TSP	1.5 sec	0.85 sec
USP	10 V	1.05 V

From comparing the results of Tables 12 and 19, one finds that the results are very similar with respect final objective function values. In most cases the constrained

TABLE 19

CASE 2 CONSTRAINED INITIAL POLE SELECTION RESULTS

Initial Poles	*Final Poles*	*Final Gains*	*Final Objective*	*Constraint Status*
-10.0 + 10.0j	-1089	25.8		X2SP
-10.0-10.0j	-189.7	5.56	.0248	Active
-300.0	-48.21	-996		
-10.0 + 10.0j	-4.08 + 29.3j	-.903		None
-10.0-10.0j	-4.08-29.3j	-.906	.1229	Active
-30.0	-10.6	-.932		
-10.0 + 10.0j	-121.7 + 130.3j	3.26		X2SP
-10.0-10.0j	-121.7-130.3j	0.439	.0299	Active
-3000.0	-44.6	-141.8		
-100.0 + 10.0j	-197.6	26.1		X2SP and
-100.0-10.0j	-1070	5.57	.0249	g_3 lower
-300.0	-47.27	-1000		Active
-10.0 + 100.0j	-61.7 + 39.3j	19.57		X2SP
-10.0-100.0j	-61.7-39.3j	7.73	.0264	Active
-300.0	-1624	-869.5		

ADS Strategy 047

case values were slightly higher, and this would be expected due to the optimum being constrained. In terms of dynamic response, however, the tightening of constraints is seen to have a considerable effect upon the resultant poles of the optimum designs.

These results indicate that the sensitivity of the final design to initial pole selection remains, in spite of the tightened constraints. It is appparent that for the Case 2 system local minima abound in the design spaces that have been investigated thus far. Restated in another way, there are many system responses, with or without overshoot, that lead to a relatively small value of the objective function. Factors that must be futher explored in an effort to find the global minimum, or at least more consistent results would be:

- investigation of more exact integration methods

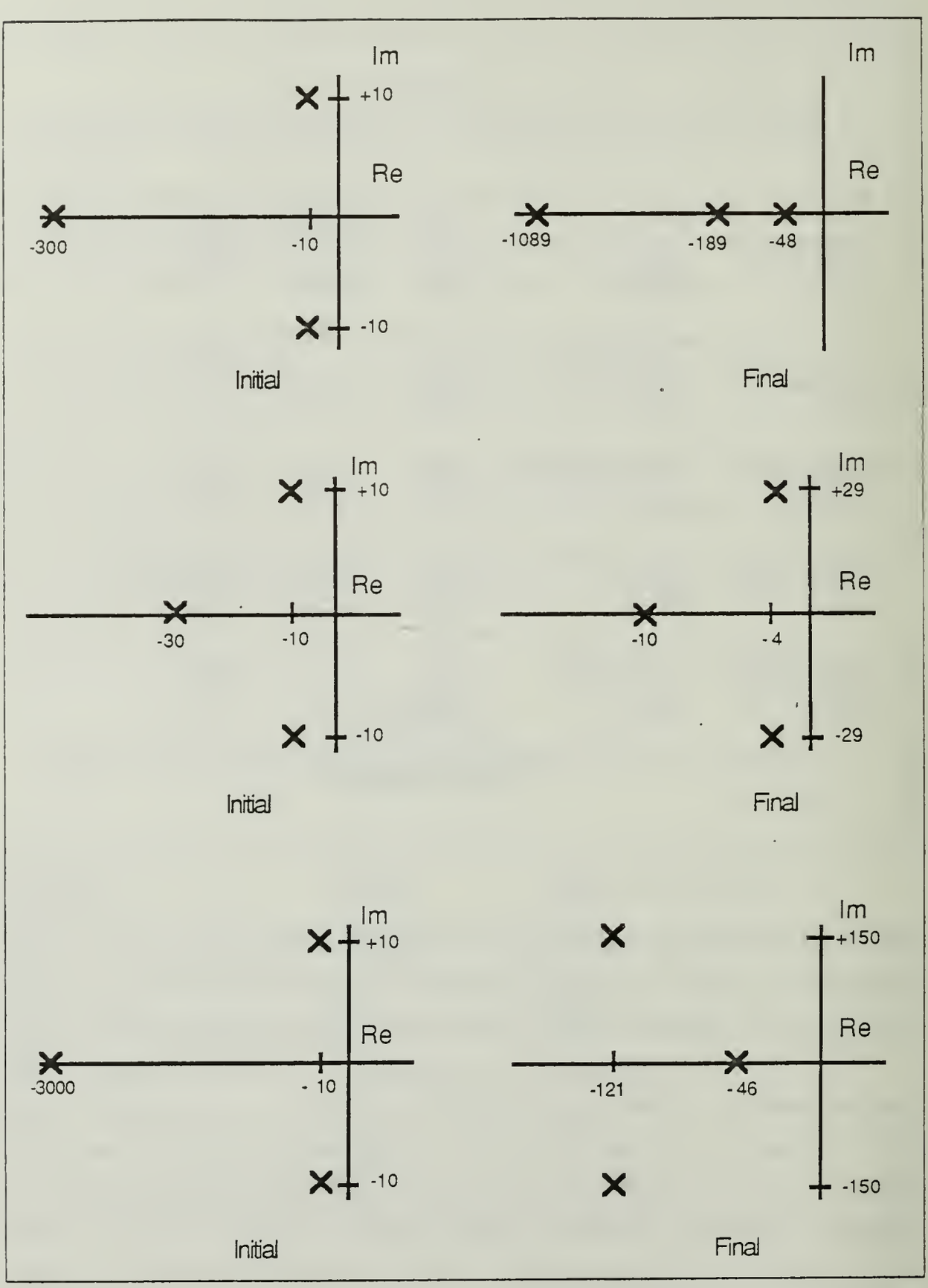


Figure 6.5a Case 2 Constrained Initial Pole Selection Results

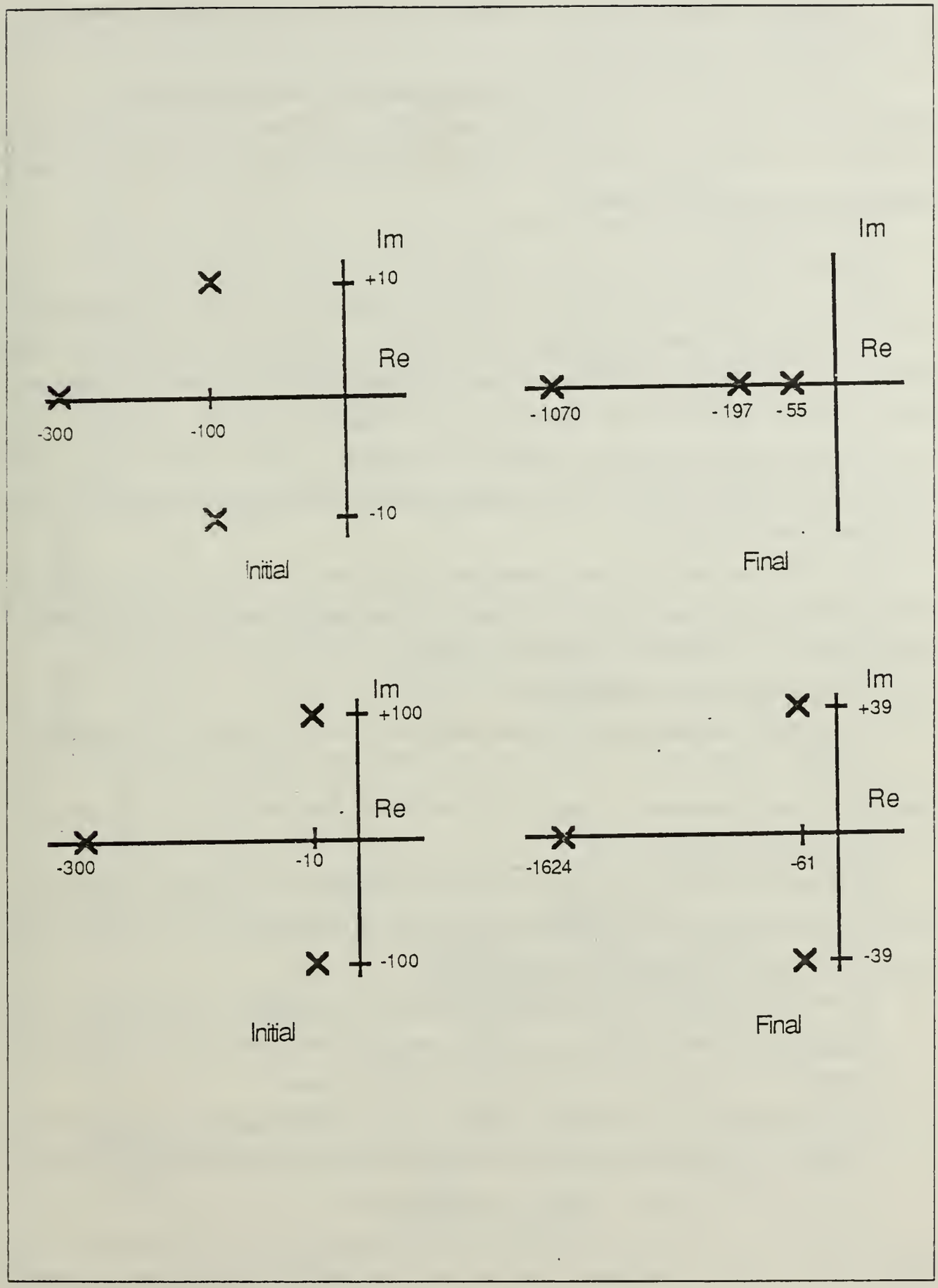


Figure 6.5b Case 2 Constrained Initial Pole Selection Results (cont.)

- increased precision of the optimization process through reduced tolerances of the ADS control parameters
- conduct a sensitivity analysis of the various optimization parameters.

Additionally, selection of a more practical application, together with known parameters associated with component hardware, would no doubt eliminate much of the design flexibilty that characterizes Case 2.

G. NOISE INPUT

The next inquiry into the Case 2 model, as describe by Tables 5 through 8 and with the maximum x_2 state constraint equal to 10.0 Amps, was to extend the initial problem to multiple inputs. This simply required changing the w_1 input from zero to a positive torque loading value, eliminating the negative response constraint, and establishing new initial and final, response and feedback control signal values. The resulting state response curves, for the assumed initial conditions and a constant torque loading, are shown in Fig. 6.6.

In order to see how the final system gains responded to different noise inputs, the constant noise torque load (w_1) of 1.0 N-m was modified to simulate a more realistic load condition. Two different signals were superimposed onto the constant load:

(1) 1000 Hz sinusoid of amplitude 0.1 N-m

(2) A random variable from a normal distribution with a standard deviation of 0.1 N-m

These inputs were generated by single calls using DSL functions and can be found in the program listed in Appendix D.

The results of these three runs are presented in Table 21. Figure 6.7 shows the final state responses for the optimized system with a constant torque loading, using strategy 133. Observations that can be drawn are:

- When the constant torque noise loading is modified, with either of the superimposed load signals, the resulting imaginary and real poles are reduced in magnitude and thus the system response will be slower.
- The final objective functions are higher in value. This would be expected as the system is further in error, due to the applied noise input, and additional energy is needed to return the system to the desired state.

Overall, the application of a noise input demonstrated the robustness of the ADSL program. By the addition of one or two simple DSL function calls, a much

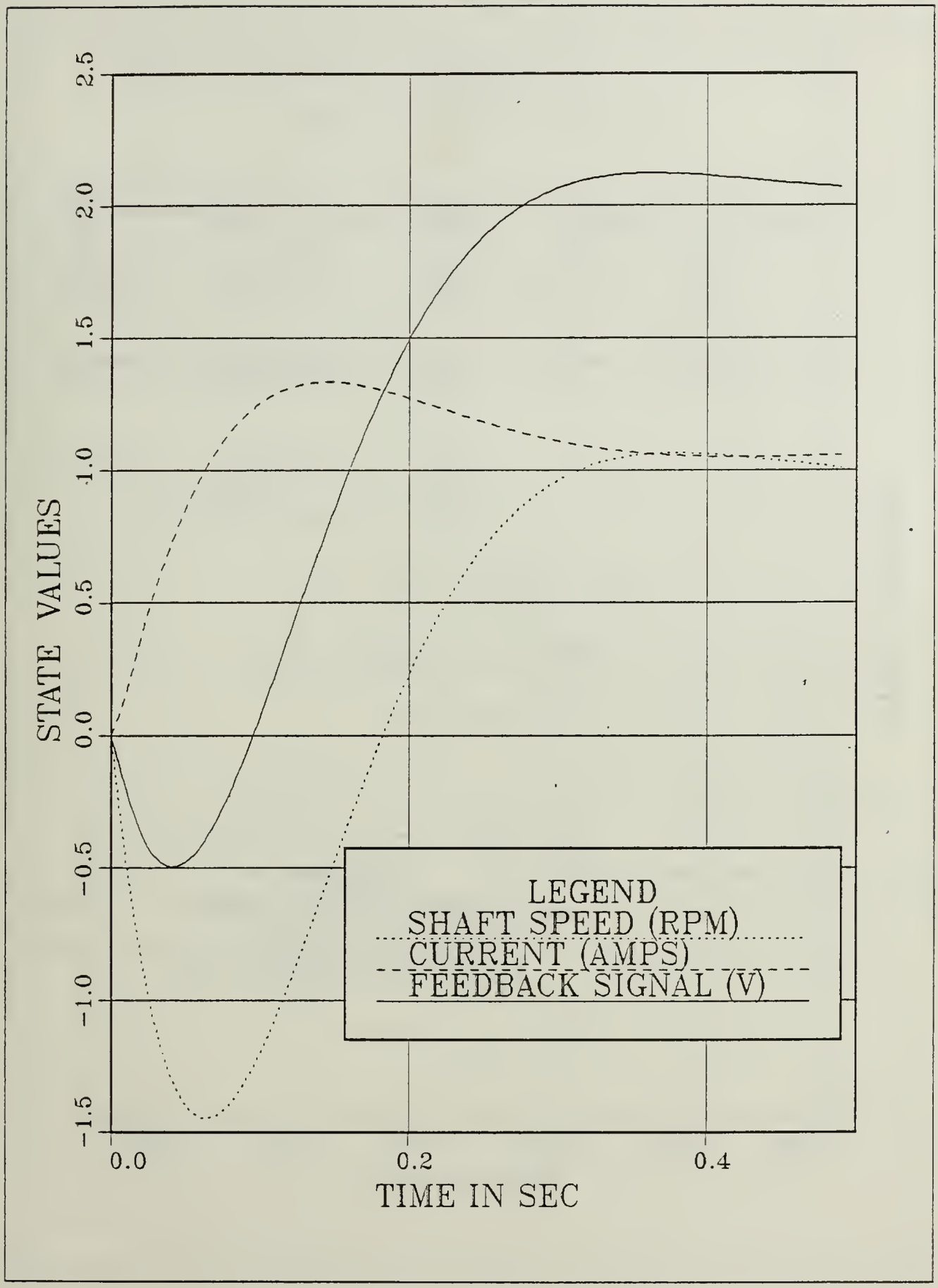


Figure 6.6 Case 2 Initial Response with Constant Torque Loading

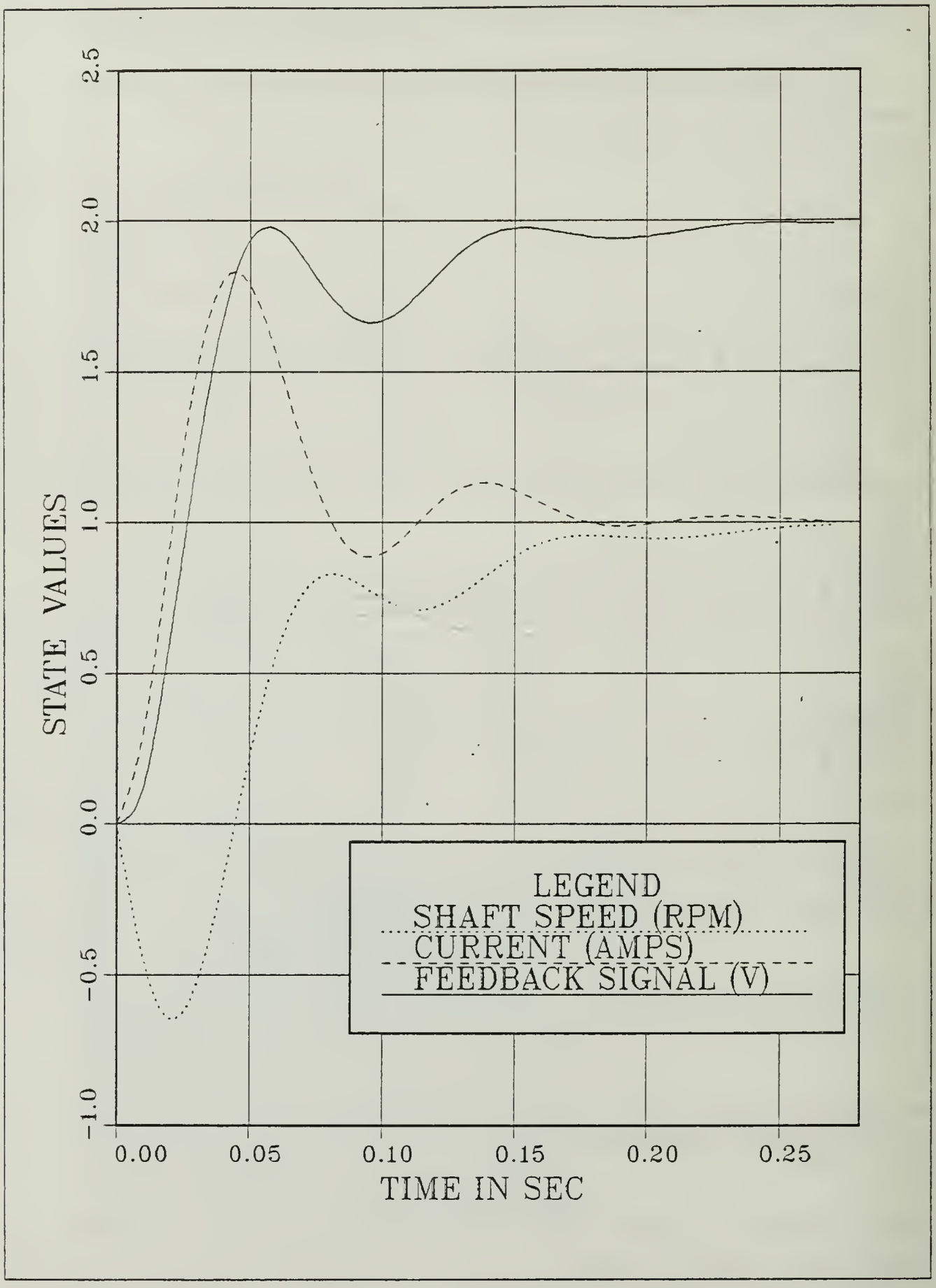


Figure 6.7 Case 2 Optimal Response with Constant Torque Loading

TABLE 21

CASE 2 NOISE INPUT RESULTS

Noise Type	*Final Poles*	*Final Gains*	*Objective*	*Total Simulations*
Constant Torque	-17.75+82.6j -17.75-82.6j -21.29	-.209 -.716 -15.2	.399	109
Normal Dist.	-9.602+25.41j -9.602-25.41j -180.6	-.579 -.001 -13.3	.484	53
Sinusoid	-9.98+3.79j -9.98-3.79j -300.8	-.379 +.603 -6.02	1.19	19

Initial Poles

-10.0+10.0j -10.0-10.0j -300.0

Initial Objective = 1.920

ADS Strategy 133

more complicated, and realistic noise input could be investigated by the control designer. Any attempt at this type of analysis, using conventional methods, would be virtually impossible.

VII. MICROCOMPUTER IMPLEMENTATION OF OPTIMIZATION FOR CONTROLLER DESIGN

A. OVERVIEW

The ADSL program, discussed in detail in the previous chapters, was run on a mainframe computer (see Appendix A for a specific description). The speed and size of this type of system makes the initial designing of an optimal controller fairly practical. The same methodology can, however, be applied to a microcomputer based program.

One system envisioned would have a microcomputer for the system controller. This computer would also monitor the running system, then using stochastic processes, generate the time averaged values of the A, B, and C matrices. These values could then be given to an ADSL like program that was running concurrently. New optimal gains would be computed, and then provided back to update the controller. Currently, work is in progress on the development of practical microcomputer controllers, but significant problems still remain. One such problem is that of obtaining sufficiently high enough sampling rates, while still being able to perform the large number of calculations required for a complex controller.

B. PROGRAM DEVELOPMENT

It was assumed that a microcomputer based integral controller, similar to that used in Case 2 could be developed, as well as the necessary A, B, and C matrix generation routines. A program structured to work like ADSL was written for the IBM Personal Computer (IBM/PC) using the Microsoft FORTRAN77 (MS-FORTRAN) compiler and linker programs [Ref. 15].

1. Microcomputer Specifics

The microcomputer selected to implement ADSLPC was the IBM/PC models XT and AT, equiped with the Intel 18087 and 180287 math co-processors respectively. This decision was based on local availibility, general acceptance as a standard machine, and speed of processing. Additionally, the degree of numerical accuracy is directly comparable with the IBM mainframe computer [Ref. 16:p. 38]. It should be pointed out that significant computational time differences were found, with the mainframe being able to run the ADSLPC program in approximately 20 seconds while 20 minutes were required using the microcomputer. The ADSLPC program generated

approximately 380 Kbytes of executable code, and this size required the PC to have 512 Kbytes of memory, and a hard disk drive for mass storage.

2. ADSLPC Coding

ADS was fully implemented using the procedures outline in the ADS Reference Manual [Ref. 4]. DSL was replaced with an independent FORTRAN main program named ADSLPC.FOR. The designation of ADSLPC was assigned to this program as a natural follow-on to the ADSL program for a personal microcomputer (PC). The five functional blocks of the ADSL program, described in Chapter 4, were closely followed in the design of the ADSLPC program. Control of time was added to the main program, as well as integration routines for solving the system of differential equations. The following programs from the International Mathematical and Statistical Library (IMSL) [Ref. 17] were used for determination of the differential equation solutions and eigenvalues:

- DGEAR
- DGRCS
- DGRPS
- DGRST
- LEQT1B
- LUDATF
- LELMF
- UERTST
- EIGRF
- EBALAF
- EBBCKF
- EHESSF
- EQRH3F

The ADS code (Version 1.12) was obtained in a format suitable for processing by the IBM/PC using the Microsoft compiler. It should be noted that not all FORTRAN compilers can be used with this version of ADS, due to the source code using FORTRAN Level 66 dynamic dimensioning standards. The IMSL routines required some slight modifications to eliminate use of unique IBM extensions to the FORTRAN 77 standard, and thus make them compatible with the Microsoft compiler. The specific changes are noted in Appendix G.

The main program was developed by modifying the translated DSL output file (FORT FORTRAN) for the Case 2 model using the eigenvalue method for the initial pole to gain determinations. The DSL generated code was first converted to single precision expressions, to be compatible with ADS, and the integration routines were replaced by a call to the IMSL subroutine DGEAR. This methology required that the derivative block of the program be broken out into its own subroutine, and that the Adams integration method be selected. Additionally, the EISPACK routine were replaced by the single precision IMSL subroutine EIGRF.

One major difference, from the ADSL program, was to incorporate the evaluation of constraints and the integration termination criteria at the end of fixed intervals (for the Case 2 model this was established at 0.1 secs), rather than at the end of every integration step. Figure 7.1 shows the interrelationships of the program calls. The complete source code is listed in Appendix H, as is a listing of the batch file used to control processing of the program.

C. COMPARATIVE RESULTS

Table 22 shows how the mainframe and PC program results compare. The general conclusions that can be drawn from these results are:

- ADSLPC can be used to find optimal controller gains for an integral feed back controller, with results comparable to those found using ADSL.
- The difference in final objective function values can be attributed to finding local minima, which can be most likely traced backed to using a different, and less accurate integration method.
- As with the ADSL model, further improvement of the numerical tolerances could result in additional reduction of the objective function.
- The obtained results compare favorably with the unscaled ADSL results found in Table 11.
- ADSLPC could be a powerful tool for optimal, on-line updating of microcomputer controller processes.

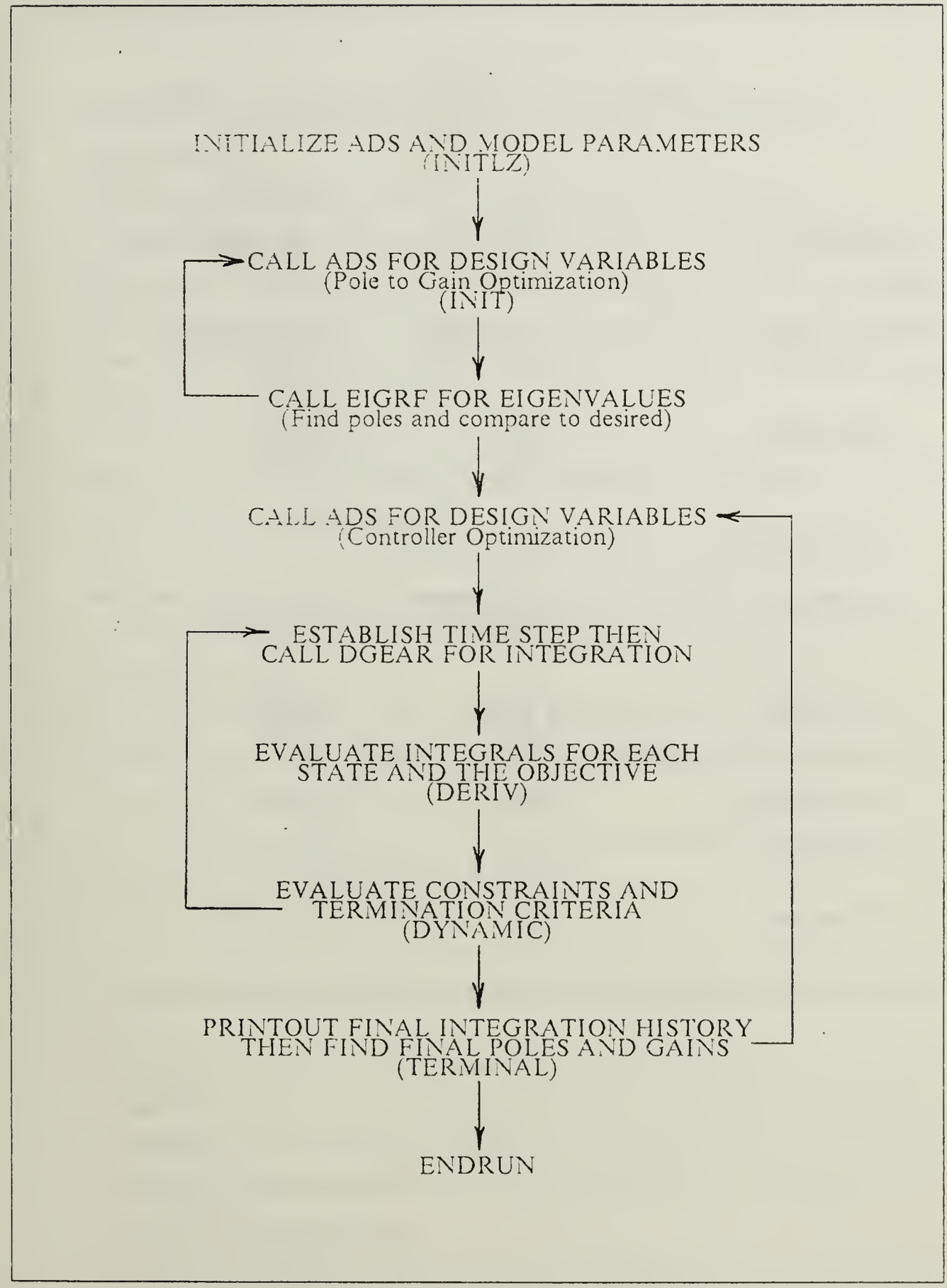


Figure 7.1 ADSL Program Segments

TABLE 22

COMPARATIVE RESULTS FOR ADSL AND ADSLPC PROGRAMS

	ADSL	*ADSLPC*
Initial Poles (desired)	-10.0 + 10.0j -10.0-10.0j -300.0	-10.0 + 10.0j -10.0-10.0j -300.0
Initial Poles (calculated)	-9.99 + 10.05j -9.99-10.05j -300.0	-9.16 + 10.00j -9.16-10.00j -299.5
Initial Gains (calculated)	-0.384 +0.599 -5.98	-0.431 +0.589 -5.71
ADS Strategy (pole to gains)	133	133
Final Poles	-79.7 + 154.7j -79.7-154.7j -47.7	-15.9 + 62.3j -15.9-62.3j -28.4
Final Gains	+0.036 +2.79 -144.2	-0.495 -0.698 -11.7
Final Objective	.0285	.0526
ADS Strategy	047	047
Evaluations	135	49

VIII. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

The following conclusions can be draw from the previous disccussions:

1. A complicated, constrained, multiple input, multiple output (MIMO) controller model can be developed using time domain methods and standard Dynamic Simulation Language (DSL) techniques. This model can then be optimized based upon a performance index of choice using the Automated Design Synthesis (ADS) program.
2. The ADSL methodology provides a robust way in which state feedback gains can be obtained without use of matrix approximations associated with the classical Riccati equation solution.
3. For MIMO systems in which sinusoidal and random noise inputs were simulated, optimal feedback gains could be found. In general they were such as to slow the system down, but closely approximated the gains found for a constant noise input.
4. The ADS optimization code, with an eigenvalue solver routine, can be employed to solve the problem of finding the system poles corresponding to given gains for any complex MIMO controller.
5. From the studies described herein, the following approaches were determined to be well suited for developing ADSL programs:
   - Use a stiff integration method
   - For initial pole to gain determination, if an exact determination is impractical, use ADS and an eigenvalue solver method as developed in this study.
   - For basic controller optimization, application of the Method of Feasible Directions strategy (047) proved to be the most practical overall, but the Sequential Unconstrained Minimization Techinque using an external penalty function strategy (133), in general, provided better optimums at the expense of more iterations.
   - Order of magnitude changes of the initial poles proved to be a reasonable method for establishing the validity of the optimization results for the final gains.

- If the final solution has active constraints, these should be investigated individually, by reduction of the corresponding constraint specification.

6. For the state feedback controller considered in this study, many local optimums can be found. The reasons for this appears to line in the definition of the performance index and the relative freedom from severe design constraints.
7. The basic ADSL program methodology can be down scaled to run on a personal computer, thus opening the possibility to using optimization in on-line applications.

B. RECOMMENDATIONS

The following recommendations are offered:

1. A simplified state feedback model should be investigated as the results obtained for the integral state feedback model demonstrated that the selected design space had many local minima. An acutal controller, with known physical parameters, would likely have a well behaved design space. This would allow for gaining a clearer understanding of the interactions investigated in this study. Additionally, the results of actual tests could then be applied to verify the ADSL predictions.
2. A large scale controller model should be optimized. One such controller model has already been developed by NASA and has been documented by Merrill [Ref. 17].
3. The methodology should be expanded to provide for online interaction, and recursive optimization capability.
4. The same methodology used with ADSL can be applied to a micro-computer based design system using trimmed down codes. One possible system would be replacement of DSL by the IBM PC-Engineering Simulation Program and ADS with the Engineering Design Optimization program Microdot, which is a reduced version of the ADS code in a format for use on a micro-computer.

APPENDIX A

ADS-DSL PROGRAM INTERFACE SPECIFICS

1. COMPUTER DESCRIPTION

The ADSL program was initially developed at the Naval Postgraduate School W. R. Church Computer Center using a IBM 3033 Attached Processor System (16 Megabytes) loosely coupled with a IBM 3033 Model U (16 Megabytes) and a IBM 4381 Model M1 (8 Megabytes). Interactive computing was provided under VM/SP CMS, and batch-processing under MVS with JES3 networking.

2. SPECIFICATIONS USED FOR COMPILING ADS (VERSION 1.10)

a. FORTRAN V/S

b. Auto-Double Precision (DBL)

c. Optimization Level 3

d. Language Level 66

3. DSL/VS (VERSION 1.1) IN THE VM MODE

The resulting TEXT file from the compilation of ADS was converted to a FORTRAN library file using the online library generation procedure [Ref. 19:pp. 35-36]. The library was named MYLIB TXTLIB in order to take advantage of the default DSL user library specification for the VM environment.

The default DSL executive was modified to streamline the interfacing of of ADS and DSL output, and is shown in Appendix B as the ADSL EXEC.

4. DSL/VS (VERSION 1.1) IN THE MVS MODE

The ADS TEXT file was also placed onto the Mass Storage System (MSS) as a user library in order that DLS/VS could access ADS using the MVS environment. The standard job control statements for using DSL must then be modified to concatenate the ADS, LINPACK, and EISPACK libraries as shown in Appendix B. The job control for compiling ADS on MSS is shown below.

5. JOB CONTROL FOR FILING ADS ON THE MASS STORAGE SYSTEM

```
//FILEADS     JOB  (0180,9999),'FILEADS',CLASS=C
//  EXEC  FORTVCL,PARM.FORT='LVL(66),AD(DBL),NOMAP,OPT(3),NOS,NOX'
//FORT.SYSIN DD *
```

```
********** REPLACE THIS LINE WITH THE ADS SOURCE CODE **************
/*
//LKED.SYSLMOD  DD UNIT=3330V,MSVGP=PUB4C,
//    DSN=MSS.F0180.ADS(ADS),
//    DISP=(NEW,CATLG,DELETE),SPACE=(CYL,(4,4,5))
```

APPENDIX B

STEPS TO RUN ADSL AT NAVAL POSTGRADUATE SCHOOL

1. VM ENVIRONMENT

a. Logon in the VM environment (note that the user must have have previously requested an increase of his default virtual machine size from 750 Kilobytes to at least 1500 Kilobytes through the Computer Center Accounting Office).

b. Define increased storage by using:

DEF STORAGE 1500K

I CMS

c. Linkto the optimization disk with the command:

LINKTO 0180P 191 OPTIM

d. Link to the DSL and MATHPACK Disks with the command:

DSLINK

This command executes the DSLINK executive procedure described below.

e. Run the ADSL program using:

ADSL FN <PLOT | TEK>

FN is the user provided filename containing the DSL input program. The filetype of the input program must be DSL. PLOT or TEK is optional, but required if any plotting is called. TEK is used when TEK-618 or IBM 3279 terminal plotting is desired, while PLOT is used for all other plotting devices. Also note that the user can not have a file name MAIN TEXT on his user disk, as this file will cause DSL to load abnormally.

f. The specified tabled output will be saved in a file named FN LISTING.

g. Plotting will then be done on the specified graphic output device and printer plots will be concatenated to the FN LISTING file.

2. DSLINK EXEC LISTING

```
EXEC LINKTO DSL
EXEC LINKTO GDDM40 191 GDDMR
EXEC LINKTO PLI5 191 PLI5
EXEC LINKTO 0062P 191 EISPACK
```

3. ADSL EXEC LISTING

```
*/* Install REXX if not here.

    &CONTROL OFF NOMSG

    EXEC REX I
    &IF &RETCODE = 0 EXEC 0 &1 &2 &3 &4 &5 &6 &7 &8 &9 &10 &11 &12 &13 &14 &15 &16 &17 &18 &19 &20
    &EXIT &RETCODE

    (end of REX installation)   */
/*********************************************************************/
/* ++++ REXX EXEC TO PERFORM COMPLETE DSL SIMULATION                 */
/*                                                                   */
/* ++++ DSL/VS SYSTEM  ...  IBM Corporation 1984                     */
/* ++++ LATEST UPDATE:  4/03/84                                      */
/*********************************************************************/
Trace value 'OFF'
/*ARG FN FT FM DISP "(" GRAF REST*/
ARG FN  GRAF
IF FN = "" THEN
TELL: Do
  Say 'The ADSL command makes a complete DSL run in the foreground, by'
  Say 'executing the following EXECs in order:  DSLT, DSLC, and DSLG.'
  Say 'The plot post-processor GRAFAEL may also be invoked at the same'
  Say 'time as an option to produce graphic output at a selected graphic'
  Say 'output device.'
  Say ' '
  SAY 'COMMAND FORMAT :  ADSL  <FN> <PLOT>  '
  Say ' '
  Say '"FN" is the filename of a previously created DSL program file.'
  Say ' '
  Say 'NOTE that the filetype must be DSL.'
  SAY ' '
  Say '"PLOT"  is an optional parameter which specifies if graphic post-'
  Say 'processing is desired.  Leaving it out, means no plotting is done.'
  SAY 'Use the command "PLOT" for printer plots or "TEK" for any other'
  SAY 'displays.'
End
IF FN = "" THEN EXIT
/* ++++ invoke DSL Translator */
      FT='DSL'
      FM='A'
      DISP='FILE'
/* GRAF='GRAFAEL'*/
'EXEC DSLT' fn ft fm
If rc >= 8 then Exit rc
COMPILE:
/* ++++ invoke Fortran Compiler */
Do
        SAY ' >>>>>>> RUNNING VS FORTRAN COMPILER <<<<<<'
End
'FORTVS FORT (LANGLVL(77) OPT(3) NOXREF NOMAP NOSDUMP NOTF NOTERM'
/*::: "FORTVS" INVOKES THE VS FORTRAN COMPILER (PROG. # 5748-F03) :::*/
/*::: Txtlibs assumed:  VALTLIB  VFORTLIB                         :::*/
IF RC >= 8 THEN
Do /* Type Error Diagnostics */
      Say ' '
      Say '*************************************************************'
      Say '        ##### ERROR(S) DETECTED IN COMPILATION PHASE #####'
      Say ' '
      Say '  Fortran listing will be displayed by XEDIT; Enter "QUIT" to exit'
      Say '*************************************************************'
      Say ' '
Queue 'PREFIX OFF'
'XEDIT FORT LISTING'
EXIT 33333
End
```

```
IF RC >=8 THEN EXIT
/* **** Test for existence of user's MYLIB TXTLIB file */
'STATE MYLIB TXTLIB'      /*::: MYLIB may be renamed :::*/
If rc                                          = 0   /*::: verify the VS Fortran Library     :::*/
                          /*::: names assumed: VALTLIB & VFORTLIB :::*/
                / * ALSO LINKTO EISPACK AND LINKPACK LIBRARIES  */
    THEN  'GLOBAL TXTLIB DSLSIM DSLSLKS VALTLIB VFORTLIB CMSLIB,'
              'EISPACK LINPACK'
    else 'GLOBAL TXTLIB MYLIB'    /*::: MYLIB may be renamed :::*/  ,
              'DSLSIM DSLSLKS VALTLIB VFORTLIB CMSLIB,'
              'EISPACK LINPACK'
/* erase old  "points" file if it exists.. */
'STATE PLOT DATA'
If rc = 0 then 'ERASE PLOT DATA'
'FILEDEF 06 DISK OUTP DATA (RECFM FBA LRECL 133 BLKSIZE 665'
'FILEDEF 05 DISK AUX DATA (RECFM FB LRECL 80 BLKSIZE 800'
'FILEDEF 12 DISK PLTC DATA (RECFM FB LRECL 80 BLKSIZE 800'
'FILEDEF 13 DISK TABL DATA (RECFM FB LRECL 80 BLKSIZE 800'
'FILEDEF 14 DISK PLOT DATA (RECFM VB BLKSIZE 1024'
'FILEDEF 15 DISK DATA DATA (RECFM FB LRECL 80 BLKSIZE 800'
'FILEDEF 16 DISK PRTSAV DATA (RECFM VBS LRECL 540 BLKSIZE 544'
'FILEDEF 21 DISK RUN DATA (RECFM FB LRECL 80 BLKSIZE 800'
/* ::: additional user file definitions you may be added here :::*/
Do
      SAY ' '
      SAY ' >>>>>>> RUNNING  DSL/VS  SIMULATOR <<<<<<<'
End
RUN:
 CP SET EMSG OFF'
'GLOBAL LOADLIB VFLODLIB'
'LOAD FORT (CLEAR RESET MAIN START NOMAP'
simrc = rc
'CP SET EMSG TEXT'
If simrc >= 8 then
Do
    Say ' '
    Say '**********************************************************************'
    Say '       ***** ERROR(S) DETECTED IN SIMULATION PHASE *****'
    Say '**********************************************************************'
    Say ' '
End
CONT:
'CP SET IMSG ON'
If simrc >= 8 then Exit 33333
IF GRAF = 'PLOT' THEN 'GRAFAEL'
IF GRAF = 'TEK' THEN 'GRAFAEL'
'ERASE 'FN ' LISTING A'
IF DISP = 'PLOT' THEN 'COPY OUTP DATA A PPLT DATA A ' FN ' LISTING A'
    ELSE 'RENAME OUTP DATA A ' FN ' LISTING A'
'X ' FN ' LISTING '
Exit   /* end of program */
```

4. SAMPLE JOB CONTROL FOR MVS ENVIRONMENT

```
//ADSL        JOB  (0180,9999),'ADSL',CLASS=B
//TEST        EXEC DSLVS,PARM.C='NOMAP,NOX,NOL,NOS,OPT(3)'
//*FORMAT PR,DDNAME=JESMSG,COPIES=0
//*FORMAT PR,DDNAME=SYSMSG,COPIES=0
//*FORMAT PR,DDNAME=JESJCL,COPIES=0
//TEST        EXEC DSLVS,PARM.C='NOMAP,NOX,NOL,NOS,OPT(3)'
//DSL.INPUT   DD   *
```

```
**** INSERT ADSL PROGRAM CODE HERE ****
END
STOP
//L.SYSLIB  DD
//          DD
//          DD
//          DD
//          DD
//          DD
//          DD DISP=SHR,DSN=MSS.F0180.ADS(ADS)
//          DD DISP=SHR,DSN=MSS.SYS3.EISPACK.LOAD
//          DD DISP=SHR,DSN=MSS.SYS3.LINPACK.LOAD
/*
//
```

APPENDIX C
EXAMPLE SISO PROGRAM LISTING

```
TITLE SISO 2ND ORDER SYSTEM-OGATA SECTION 7-3 WITH CONSTRAINTS

**************************************************************************
* LISTING OF ALL VARIABLES
*
*  A       WORK MATRIX FOR ADS
*  DF      VECTOR OF DESIGN VARIABLE GREADIENTS
*  G       VECTOR OF CONSTRAINT VALUES
*  H2      VALUE OF THE RESPONSE OSCILATION
*  I       LOOP COUNTER
*  IC      VECTOR OF CONSTRAINT GRADIENTS
*  IDG     VECTOR OF CONSTRAINT TYPES
*  IGRAD   SCALAR FOR INDICATING METHOD FOR OBTAINING GRADIENTS
*  INFO    SCALAR INDICATING OPTIMIZATION STATUS
*  IONED   SCALAR INDICATING TYPE OF ONE-DIMESIONAL SEACH
*  IOPT    SCALAR INDICATING TYPE OF OPTIMIZATION
*  IPRINT SCALAR INDICATING TYPE OF PRINT CONTROL FOR ADS
*  ISTRAT SCALAR INDICATING TYPE OF STRATEGY
*  IWK     INTEGER WORK VECTOR FOR ADS
*  LSTOBJ VALUE OF THE OBJECTIVE FUNCTION AT PREVIOUS INTEGRATION STEP
*  LSTTIM VALUE OF TIME AT THE PREVIOUS INTEGRATION STEP
*  NCOLA   SCALAR COLUMN DIMENSION OF A MATRIX
*  NCON    NUMBER OF CONSTRAINTS
*  NDV     NUMBER OF DESIGN VARIABLES
*  NGT     SCALAR ADS CONTROL VARIABLE
*  NRA     SCALAR ROW DIMENSION OF A MATRIX
*  NRIWK   SCALAR DIMENSION OF IWK VECTOR
*  NRWK    SCALAR DIMENSION OF WK VECTOR
*  OBJ     OBJECTIVE FUNCTION VALUE
*  R       VALUE OF THE INPUT TO THE SYSTEM
*  TAU     SYSTEM TIME CONSTANT
```

```
*  TIME    DSL TIMER VALUE
*  VLB     VECTOR OF DESIGN VARIABLE LOWER BOUNDS
*  VUB     VECTOR OF DESIGN VARIABLE UPPER BOUNDS
*  WK      ADS WORK VECTOR
*  X       VECTOR OF DESIGN VARIABLES
*  Y       VALUE OF THE SYSTEM OUTPUT
*  YMAX    MAXIMUM VALUE OF THE SYSTEM OUTPUT
*  YOSC    VECTOR OF SYSTEM OUTPUT VALUES AT PREVIOUS INTEGRATION STEPS
************************************************************************

* PREAMBLE TO INITIALIZE THE ADS PARAMETERS
FIXED ISTRAT,IOPT,IONED,IPRINT,INFO,IGRAD
FIXED NDV,NCON,IDG,NGT,IC,NRA,NCOLA,NRWK,IWK,NRIWK,I

* DIMENSIONING REQUIREMENTS ARE FOUND IN THE ADS MANUAL TABLE 5
* ARRAYS MUST BE ASSIGNED USING THE DIMENSION STATEMENT IN DSL
D     DIMENSION A(2,6)
* VECTORS CAN BE ASSIGNED USING THE ARRAY STATEMENT IN DSL
ARRAY WK(1000),IWK(500),DF(2)
ARRAY X(2),VLB(2),VUB(2),G(5),IDG(5),IC(5)

* INITIALIZATION OF ADS INPUT CONSTANTS (ADS MANUAL TABLE 5)
PARAM NRA=1,NCOLA=1,NRWK=1000,NRIWK=500
PARAM IGRAD=0,INFO=0,NDV=1,NCON=5

* SETTING ALL CONSTRAINTS TO NON-LINEAR, INEQUALITY
TABLE IDG(1-5)=0,0,0,0,0

* INITIAL GUESS FOR DESIGN VARIABLE ZETA
TABLE X(1)=1.

* UPPER AND LOWER BOUNDS FOR DESIGN VARIABLE
TABLE VLB(1)=.01 , VUB(1)=2.0

* SET ADS OPTIMIZATION STRATEGY  (ADS MANUAL PART 4.6)
PARAM ISTRAT=1, IOPT=3, IONED=3, IPRINT=2020

*SET ADS PRINT PARAMETER (ADS MANUAL TABLE 5)
```

```
PARAM IPRINT=1000

* SET DSL INTEGRATION OUTPUT PARAMETERS (DSL MANUAL PART 4.2)
METHOD STIFF
CONTROL FINTIM=45
SAVE Y,OBJ
GRAPH (DE=TEK618) TIME,Y,OBJ
PRINT Y,OBJ

*****************************************************************

INITIAL

* CALL TO ADS TO GET THE DESIGN VARIABLE FOR THE SYSTEM
* CALLING STATEMENT SPECIFIED IN ADS MANUAL PART 4.0
         CALL ADS(INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,...
                  NDV,NCON,X,VLB,VUB,OBJ,G,IDG,NGT,IC,...
                  DF,A,NRA,NCOLA,WK,NRWK,IWK,NRIWK)

* THIS STATEMENT TURNS ON THE PRINTER OUTPUT ONLY ON THE LAST RUN
* WHEN INFO HAS BEEN RESET TO 0 BY ADS TO INDICATE OPTIMIZATION
* HAS BEEN COMPLETED
         IF (INFO.EQ.0) DELPRT=1.5
*THIS STATEMENT TURNS ON THE RECORDING OF PLOTTER VALUES
         IF (INFO.EQ.0) DELPLT=.4

* RE-INITIALIZE CONSTRAINT CHECK VALUES AFTER CALL TO ADS
         YMAX=0.
         TAU=0.
         LSTOBJ=0.
         LSTTIM=0.
         YOSC1=0.
         YOSC2=0.
         YOSC3=0.
         DO 15 I=1,NCON
      15 G(I)=1.D20

*****************************************************************
```

```
DERIVATIVE
NOSORT

* INPUT TO THE MODEL (DSL MANUAL SECTION 5.1)
        R=STEP(0.)

* OUTPUT OF SECOND ORDER LAG SYSTEM (DSL MANUAL SECTION 5.4)
        Y=CMPXPL(0.,0.,X(1),1.,R)

* ITAE PERFORMANCE INDEX TO MINIMIZE AMPLITUDE (OGATA P. 299)
        OBJ=INTGRL(0.,TIME*ABS(R-Y))

************************************************************

DYNAMIC

* UPDATE CONSTRAINT CHECK VALUES AT EACH INTEGRATION STEP
        YOSC1=YOSC2
        YOSC2=YOSC3
        YOSC3=Y
        H2=ABS(YOSC2-R)
        IF(TAU.NE.0.)GOTO 20
        IF(Y.GE.(0.63212*R))TAU=TIME
    20 IF(Y.GT.YMAX)  YMAX=Y

* MAX AMPLITUDE CONSTRAINT ( <140%)
        IF(Y.EQ.YMAX) G(1)=YMAX-(1.4*R)

* MAX DELAY TIME CONSTRAINT ( <1.5 sec)
        IF((Y.LE.(.5*R)).AND.(Y.EQ.YMAX)) G(2)=TIME-1.5

* MAX RISE TIME CONSTRAINT ( <3.5 sec)
        IF((Y.LE.R).AND.(Y.EQ.YMAX)) G(3)=TIME-3.5

* MAX PEAK TIME CONSTRAINT ( <4.5 sec)
        IF(Y.EQ.YMAX) G(4)=TIME-4.5

* MAX COMPLETION TIME CONSTRAINTS ( <12 sec)
        IF(G(5).LT.0.)GOTO 30
* FOR UNDER DAMPED CASES
```

```
          IF((H2.GT.ABS(YOSC1-R)).AND.(H2.GT.ABS ...
          (YOSC3-R)).AND.(H2.LE.(.01*R))) G(5)=TIME-12.
* FOR OVER DAMPED AND CRITICALLY DAMPED CASES
          IF(((Y-R).LT.(.01*R)).AND.(TIME.GE.(5.*TAU))...
          .AND.(TAU.NE.0.).AND.(Y.EQ.YMAX)) G(5)=TIME-12.

* INTEGRATION COMPLETE CHECK USING THE OBJECTIVE FUNCTION SLOPE
* CHECK FOR FIST INTEGRATION STEP
     30 IF((OBJ.LE.0.).OR.(LSTOBJ.LE.0.))GOTO 40
* DETERMINE SLOPE AND CHECK FOR PAST COMPLETION TIME - IF MET
* ENDRUN DIRECTS PROGRAM TO THE TERMINAL SEGMENT OTHERWISE THE
* NEXT INTEGRATION STEP IS TAKEN UNTIL FINTIM IS MET
          IF(((OBJ-LSTOBJ)/(TIME-LSTTIM).LT.1.E-5).AND. ...
          (((H2.LE.(.01*R)).AND.(H2.GT.ABS(YOSC1-R))  ...
          .AND.(H2.GT.ABS(YOSC3R))).OR.((TIME.       ...
          GE.(5.*TAU)).AND.(TAU.NE.0.))))CALL ENDRUN
     40 LSTOBJ=OBJ
          LSTTIM=TIME
**********************************************************************

TERMINAL

* CHECK TO OPTIMIZATION COMPLETE TO EXIT DSL OTHERWISE RERUN WILL
* START THE PROGRAM OVER AT THE INITIAL SEGMENT
          IF (INFO.EQ.0) CALL ENDJOB
          CALL RERUN

END

STOP
```

APPENDIX D

STATE VARIABLE PROGRAM LISTINGS

```
TITLE DC MOTOR INTEGRAL CONTROL SYSTEM-NO NOISE-KUO EXAMPLE 8-7
*************************************************************************
* LISTING OF ALL VARIABLES
*
* AA      MATRIX OF STATE COEFFICIENTS
* BB      VECTOR OF CONTROL COEFFICIENTS
* C       SCALAR OUTPUT VALUE
* CC      MATRIX OF STATE OUTPUT COEFFICEINTS
* CF      FINAL EXPECTED OUTPUT VALUE
* CI      INITIAL OUTPUT VALUE
* CON     VECTOR OF CONSTRAINT VALUES
* CSP     MAXIMUM PERCENT OVERSHOOT FOR THE OUTPUT
* DELC    DIFFERENCE OF THE ACTUAL OUTPUT TO THE INITIAL OUTPUT
* DELU    DIFFERENCE OF THE ACTUAL CONTROL TO THE INITIAL CONTROL
* DELX2   DIFFERENCE OF THE ACTUAL X2 STATE TO THE INITIAL X2 STATE
* DV      VECTOR OF DESIGN VARIABLE GREADIENTS
* F       VECTOR OF INPUT COEFFICIENTS
* FV1     EISPAC WORK VECTOR
* G       VECTOR OF DESIGN VARIABLE GAIN VALUES
* H       VECTOR OF OUTPUT COEFFICEINTS FOR THE INPUTS
* I       LOOP COUNTER
* IC      VECTOR OF CONSTRAINT GRADIENTS
* IDG     VECTOR OF CONSTRAINT TYPES
* IER1    LINPAC ERROR INDICATOR
* IERR    EISPAC ERROR INDICATOR
* IGRAD   SCALAR FOR INDICATING METHOD FOR OBTAINING GRADIENTS
* II      COUNTER OF NUMBER OF FUNCTION EVALUATIONS
* INFO    SCALAR INDICATING OPTIMIZATION STATUS
* IONED   SCALAR INDICATING TYPE OF ONE-DIMESIONAL SEACH
* IOPT    SCALAR INDICATING TYPE OF OPTIMIZATION
* IPRINT  SCALAR INDICATING TYPE OF PRINT CONTROL FOR ADS
```

```
* IFVT   LINPAC INTEGER WORK VECTOR
* ISTRAT SCALAR INDICATING TYPE OF STRATEGY
* IV1    EISPAC INTEGER WORK VECTOR
* IWK    INTEGER WORK VECTOR FOR ADS
* J      LOOP COUNTER
* LSLOPE SLOPE OF THE OBJECTIVE FUNCTION FOR THE LAST INTEGRATION INTERVAL
* LSTOBJ VALUE OF THE OBJECTIVE FUNCTION FOR THE LAST INTEGRATION INTERVAL
* LSTTIM VALUE OF TIME AT THE PREVIOUS INTEGRATION STEP
* MATZ   EISPAC SCALAR VALUE TO INDICATOR EIGENVALUE DETERMINATION
* MEAN   DSL NORMAL DISTRIBUTION MEAN
* NCOLA  SCALAR COLUMN DIMENSION OF WA MATRIX
* NCON   NUMBER OF CONSTRAINTS
* NDV    NUMBER OF DESIGN VARIABLES
* NGT    SCALAR ADS CONTROL VARIABLE
* NM     EISPAC DIMENSION OF MATRIX SIZE FOR EIGENVALUE DETERMIANTION
* NN     EISPAC DIMENSION OF MATRIX SIZE FOR EIGENVALUE DETERMIANTION
* NRA    SCALAR ROW DIMENSION OF WA MATRIX
* NRIWK  SCALAR DIMENSION OF IWK VECTOR
* NRWK   SCALAR DIMENSION OF WK VECTOR
* OBJ    OBJECTIVE FUNCTION VALUE
* OOBJ   OBJECTIVE FUNCTION FOR INTEGRATION
* P1     DESIRED INITIAL COMPLEX POLE VALUE
* P2     DESIRED INITIAL COMPLEX POLE VALUE
* P3     DESIRED INITIAL COMPLEX POLE VALUE
* PA     MATRIX OF INITIAL GAIN DETERMINATION COEFFICIENTS
* Q      STATE WEIGHTING MATRIX
* R      CONTROL WEIGHTING VECTOR
* SD     STANDARD DEVIATION OF DSL NORMAL FUNCTION
* SEED   RANDOM MUMBER SEED FOR DSL NORMAL FUNCTION
* TAU    SYSTEM TIME CONSTANT
* TC     TIME WHEN COMPLETION TIME SPECIFICATION IS MET
* TIME   DSL TIMER VALUE
* TSP    MAXIMUM COMPLETION TIME
* U      SCALAR FEEDBACK CONTROL VALUE
* UI     INITIAL VALUE OF THE CONTROL VECTOR
```

```
* UMAX   MAXIMUM CONTROL VALUE
* USP    MAXIMUM VALUE OF THE CONTROL VECTOR
* UT     TRANSLATED FEEDBACK CONTROL VALUE
* V      MATRIX FOR FINAL POLE DETERMINATION USING EISPACK
* VLB    VECTOR OF DESIGN VARIABLE LOWER BOUNDS
* VUB    VECTOR OF DESIGN VARIABLE UPPER BOUNDS
* W1     DISTURBANCE VALUE
* W2     REFERENCE SETTING
* WA     ADS WORK VECTOR
* WI     VECTOR OF FINAL IMAGINARY POLES
* WL     CIRCULAR FREQUENCY OF SINUSOIDAL DISTRUBANCE
* WR     VECTOR OF FINAL REAL POLES
* X1     MOTOR SHAFT SPEED STATE VALUE
* X1T    TRANSLATION OF X1 STATE
* X2     MOTOR CURRENT STATE VALUE
* X2I    INITIAL VALUE OF THE X2 STATE
* X2MAX  MAXIMUM VALUE OF THE X2 STATE
* X2SP   MAXIMUM SPECIFIED VALUE OF THE X2 STATE
* XX1    FUNCTION DESCRIBING THE X1 STATE FOR INTEGRATION
* XX2    FUNCTION DESCRIBING THE X2 STATE FOR INTEGRATION
* YOSC1  VALUE OF SYSTEM OUTPUT OSCILATION AT TWO PREVIOUS INTEGRATION STEPS
* YOSC2  VALUE OF SYSTEM OUTPUT OSCILATION AT PREVIOUS INTEGRATION STEP
* YOSC3  VALUE OF SYSTEM OUTPUT OSCILATION
* Z      EISPACK DUMMY SCALAR
*************************************************************************
* INITIALIZE THE ADS PARAMETERS
FIXED NRA,NGT,NCOLA,NRWK,NRIWK,IGRAD,NDV,NCON,IDG,IC,IWK
FIXED ISTRAT,IOPT,IONED,IPRINT,INFO,IPVT
ARRAY DV(4),VLB(4),VUB(4),CON(6),IDG(6),IC(6),DF(4),WK(10DD),IWK(1D0D)
D     DIMENSION WA(4D,4D)
PARAM NRA=4D,NCOLA=40,NRWK=1000,NRIWK=1DD0,NDV=3,NCON=6,IGRAD=D,INFO=D
* ESTABLISH TYPES OF CONSTRAINTS (ADS MANUAL TABLE 5)
TABLE IDG(1-6)=3*D,2*2,D
* LOWER BOUND FOR DESIGN VARIABLES
TABLE VLB(1-3)=3*-1DDD.
```

```
* UPPER BOUND FOR DESIGN VARIABLES
TABLE VUB(1-3)=3*1000.
* ADS OPTIMIZATION METHOD SELECTION
PARAM ISTRAT=0, IOPT=4, IONED=7, IPRINT=1010
*************************************************************************
* INITIALIZE THE DSL PARAMETERS
PRINT OBJ,C,X1,X2,U
SAVE (RUN) OBJ,X1,X2,U,C
CONTROL DELPLT=0.005,DELPRT=0.01
CONTROL FINTIM=4.
METHOD STIFF
RELERR OBJ=1.E-4
ABSERR OBJ=1.E-4
*************************************************************************
* INITIALIZE THE MODEL PARAMETERS
FIXED I,II,IER1,IERR,NN,NM,IV1,MATZ,SEED
COMPLEX P1,P2,P3
ARRAY BB(3),H(2),R(2),IPVT(3),G(3),WR(3),WI(3),IV1(3),FV1(3)
D     DIMENSION AA(3,3),CC(2,2),Q(2,2),PA(3,3),F(2,2),V(3,3),Z(3,3)
* ZERO ALL MATRICES
D     DATA AA,F,CC,Q/21*0./
TABLE BB(1-3)=3*0.,R(1-2)=3*0.,H(1-2)=3*0.
* SET CONSTRAINT SPECIFICATIONS
* CI=INITIAL OUTPUT VALUE;  CF=FINAL EXPECTED OUTPUT VALUE
* CSP=MAXIMUM PERCENT OVERSHOOT FOR THE OUTPUT
* TSP=MAXIMUM COMPLETION TIME
* USP=MAXIMUM VALUE OF THE CONTROL VECTOR
PARAM CI=1.,CF=0.,CSP=5.,TSP=1.00,USP=1.05,X2SP=1.0
* SET INITTAL INPUT VALUES (W1=DISTURBANCE, W2=REFERENCE SETTING)
PARAM W1=0., W2=1., SEED=113, MEAN=1., SD=0.1, WL=157
* SET INITIAL RUN FLAG AND EISPACK PARAMETERS
PARAM II=0,MATZ=0,NM=3,NN=3,I=1,J=1,IER1=0,IERR=0
*************************************************************************
INITIAL
* CHECK FOR INITIAL RUN
```

```
      IF(II.GT.0) GOTO 200
* THE DESIGN VARIABLES ARE THE FEEDBACK GAINS
* INITIAL SYSTEM POLES TO BASE FEEDBACK GAINS UPON (COMPLEX FORM)
      P1=CMPLX(-10.0,10.0)
      P2=CMPLX(-10.0,-10.0)
      P3=CMPLX(-300.0,0.)
* SET NON-ZERO COEFFICIENTS
      AA(1,2)=50.
      AA(2,1)=-200.
      AA(2,2)=-200.
      AA(3,1)=-1.
      BB(2)=200.
      CC(1,1)=-1.
      H(2)=1.
      F(1,1)=-50.
      Q(1,1)=1.
      Q(2,2)=1.
      R(1)=1.
* ROUTINE TO FIND THE SYSTEM GAINS FROM KNOWN POLES (KUO SECTION 8-6)
      G(1)=REAL(-P1-P2-P3)+AA(1,1)+AA(2,2)+AA(3,3)
      G(2)=REAL(P1*P2+P1*P3+P2*P3)-AA(1,1)*AA(2,2)-...
      AA(2,2)*AA(3,3)-AA(1,1)*AA(3,3)+AA(2,3)*AA(3,2)+AA(1,2)* ...
      AA(2,1)+AA(1,3)*AA(3,1)
      G(3)=REAL(-P1*P2*P3)+AA(1,1)*AA(2,2)*AA(3,3)-AA(1,1)*...
      AA(2,3)*AA(3,2)+AA(1,2)*AA(2,3)*AA(3,1)-AA(1,2)*AA(2,1)*...
      AA(3,3)+AA(1,3)*AA(2,1)*AA(3,2)-AA(1,3)*AA(3,1)*AA(2,2)
      PA(1,1)=BB(1)
      PA(1,2)=BB(2)
      PA(1,3)=BB(3)
      PA(2,1)=(-AA(2,2)-AA(3,3))*BB(1)+AA(1,2)*BB(2)+AA(1,3)*BB(3)
      PA(2,2)=AA(2,1)*BB(1)+(-AA(3,3)-AA(1,1))*BB(2)+AA(2,3)*BB(3)
      PA(2,3)=AA(3,1)*BB(1)+AA(3,2)*BB(2)+(-AA(2,2)-AA(1,1))*BB(3)
      PA(3,1)=(AA(2,2)*AA(3,3)-AA(2,3)*AA(3,2))*BB(1)-(AA(1,2)* ...
      AA(3,3)-AA(1,3)*AA(3,2))*BB(2)+(AA(1,2)*AA(2,3)-AA(1,3)* ...
      AA(2,2))*BB(3)
```

```
      PA(3,2)=-(AA(2,1)*AA(3,3)-AA(2,3)*AA(3,1))*BB(1)+(AA(1,1)* ...
      AA(3,3)-AA(1,3)*AA(3,1))*BB(2)-(AA(1,1)*AA(2,3)-AA(1,3)* ...
      AA(2,1))*BB(3)
      PA(3,3)=(AA(2,1)*AA(3,2)-AA(3,1)*AA(2,2))*BB(1)-(AA(1,1)* ...
      AA(3,2)-AA(1,2)*AA(3,1))*BB(2)+(AA(1,1)*AA(2,2)-AA(1,2)* ...
      AA(2,1))*BB(3)
      WRITE(6,297)P1,P2,P3
297   FORMAT('0 INITIAL POLES ARE : ',3('(',E11.4,',',E11.4,' J)  '))
      CALL DGEFA(PA,3,3,IPVT,IER1)
      IF(IER1.NE.0)GOTO 300
      CALL DGESL(PA,3,3,IPVT,G,0)
      WRITE(6,299)G(1),G(2),G(3)
299   FORMAT('0 INITIAL GAINS G(1),G(2),G(3) = ',3E15.4)
      WRITE(6,290)
 290  FORMAT('0 STATE CONTROL MATRIX (B  AB) = ')
      WRITE(6,298) BB(1),AA(1,1)*BB(1)+AA(1,2)*BB(2)
      WRITE(6,298) BB(2),AA(2,1)*BB(1)+AA(2,2)*BB(2)
 298  FORMAT(2E15.4)
      WRITE(6,291)
 291  FORMAT('0 OUTPUT CONTROL MATRIX (CB  CAB) = ')
      WRITE(6,298) CC(1,1)*BB(1)+CC(1,2)*BB(2),CC(1,1)*(AA(1,1)* ...
      BB(1)+AA(1,2)*BB(2))+CC(1,2)*(AA(2,1)*BB(1)+AA(2,2)*BB(2))
      WRITE(6,292)
 292  FORMAT('0 OBSERVABILITY MATRIX (C*  A*C*) = ')
      WRITE(6,298) CC(1,1),AA(1,1)*CC(1,1)+AA(2,1)*CC(1,2)
      WRITE(6,298) CC(1,2),AA(1,2)*CC(1,1)+AA(2,2)*CC(1,2)
      GOTO 302
300   WRITE(6,301) IER1
301   FORMAT('0 POLE SELECTION ERROR IER1 = ',I7)
      CALL ENDJOB
302   CONTINUE
      DV(1)=G(1)
      DV(2)=G(2)
      DV(3)=G(3)
      IF(INFO.EQ.0) GOTO 200
```

```
      CALL ADS(INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,DV,VLB,...
     VUB,OBJ,CON,IDG,NGT,IC,DF,WA,NRA,NCOLA,WK,NRWK,IWK,NRIWK)
      IWK(2)=0
      WRITE(6,220)
  220 FORMAT('0      SCALING NOT USED')
* CALL TO ADS TO GET THE DESIGN VARIABLE FOR THE SYSTEM
  200 CALL ADS(INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,DV,VLB,...
     VUB,OBJ,CON,IDG,NGT,IC,DF,WA,NRA,NCOLA,WK,NRWK,IWK,NRIWK)
      G(1)=DV(1)
      G(2)=DV(2)
      G(3)=DV(3)
* CHECK FOR TYPE OF RUN IN ORDER TO SAVE INITIAL AND FINAL RESULTS
      IF(II.NE.0)GOTO 235
GRAPH (INITIAL/RUN,RU=1,DE=TEK618 ) TIME,X1,X2,U
      GOTO 245
  235 IF(INFO.NE.0)GOTO 240
      DELPRT=0.01
      DELPLT=0.01
GRAPH (FINAL/RUN,RU=*,DE=TEK618 ) TIME,X1,X2,U
      GOTO 245
  240 DELPRT=0.0
      DELPLT=0.0
  245 II=1
* INITIALIZE CONSTRAINT CHECK VALUES
* LAST THREE VALUES OF THE OUTPUT
      YOSC1=CI
      YOSC2=CI
      YOSC3=CI
* MAXIMUM VALUE OF THE OUTPUT
      CMAX=0.
* THE TIME CONSTANT OF THE OUTPUT
      TAU=D.
* TIME WHEN COMPLETION TIME SPECIFICATION IS MET
      TC=0.
* INITIAL VALUE OF THE CONTROL VECTOR
```

```
        UI=0.
* MAXIMUM VALUE OF THE CONTROL
        UMAX=0.
* INITIAL VALUE OF THE X2 STATE
        X2I=0.
* MAXIMUM VALUE OF THE X2 STATE
        X2MAX=0.
* VALUE OF THE OBJECTIVE FUNCTION FOR THE LAST INTEGRATION INTERVAL
        LSTOBJ=0.
* SLOPE OF THE OBJECTIVE FUNCTION FOR THE LAST INTEGRATION INTERVAL
        LSLOPE=0.
* VALUE OF TIME AT END OF LAST INTEGRATION INTERVAL
        LSTTIM=0.
**********************************************************************
DERIVATIVE
* PERFORM INTEGRATIONS TO FIND NEW STATES
          X1=INTGRL(0.,XX1)
          X2=INTGRL(0.,XX2)
          XX1=AA(1,1)*X1+AA(1,2)*X2+BB(1)*U+F(1,1)*W1+F(1,2)*W2
          XX2=AA(2,1)*X1+AA(2,2)*X2+BB(2)*U+F(2,1)*W1+F(2,2)*W2
* FIND NEW CONTROL VALUE
          U=-G(1)*X1-G(2)*X2-G(3)*INTGRL(0.,C)
* FIND NEW OUTPUT VALUE
          C=CC(1,1)*X1+CC(1,2)*X2+H(1)*W1+H(2)*W2
* FIND NEW OBJECTIVE FUNCTION VALUE USING QUADRATIC PERFORMANCE INDEX
* AFTER TRANSLATION OF OUTPUT AND CONTROL VALUES
          X1T=X1-1.
          UT=U-1.
          OBJ=INTGRL(0.,OOBJ)
          OOBJ=(X1T*Q(1,1)+X2*Q(1,2))*X1T+(X1T*Q(2,1)+X2*Q(2,2))*X2+ ...
          UT*R(1)*UT
**********************************************************************
DYNAMIC
* DISTURBANCE FUNCTION
*     W1=NORMAL(SEED,MEAN,SD)
```

```
*     W1=1.+0.1*SIN(WL*TIME)
* CONSTRAINT CHECKS AT EACH INTEGRATION STEP
* ESTABLISH MAX OUTPUT OSCILLATION CHECKS (YOSC)
* STEP FORWARD THE LAST THREE OUTPUT VALUES
          YOSC1=YOSC2
          YOSC2=YOSC3
          YOSC3=C
* CALCULATE THE DIFFERENCE OF THE ACTUAL OUTPUTS TO THE DESIRED OUTPUTS
          H1=H2
          H2=H3
          H3=ABS(YOSC3-CF)
* CALCULATE THE DIFFERENCE OF THE ACTUAL OUTPUT TO THE INITIAL OUTPUT
          DELC=ABS(CI-C)
* CALCULATE THE DIFFERENCE OF THE ACTUAL CONTROL TO THE INITIAL CONTROL
          DELU=ABS(UI-U)
* FIND MAXIMUM CONTROL VALUE
          IF(DELU.GE.UMAX)UMAX=DELU
* CALCULATE THE DIFFERENCE OF THE ACTUAL CONTROL TO THE INITIAL CONTROL
          DELX2=ABS(X2I-X2)
* FIND MAXIMUM X2 STATE VALUE
          IF(DELX2.GE.X2MAX)X2MAX=DELX2
* ESTABLISH TIME CONSTANT CHECK (TAU)
          IF(TAU.NE.0.)GOTO 20
          IF(DELC.GE.ABS(0.63212*(CI-CF)))TAU=TIME
       20 IF(DELC.GE.CMAX)  CMAX=DELC
* MAX AMPLITUDE CONSTRAINT (CON(1))
* (+) IF ABOVE SPECIFICATION    (-) IF BELOW SPECIFICATION
          IF(DELC.EQ.CMAX) CON(1)=DELC-ABS(CI-CF)*(CSP/100+1.)
* MAX COMPLETION TIME CONSTRAINT (CON(2))
* SKIP CHECK IF COMPLETION TIME ALREADY MET
          IF(TC.NE.0.)GOTO 150
* FOR UNDER DAMPED CASE
* CHECK FOR A OSCILLATION REVERSAL, AND IF THE PEAK MAGNITUDE IS LESS
* THAN 1% OF THE DIFFERENCE BETWEEN THE INITIAL AND FINAL OUTPUT VALUES
          IF((H2.GE.H1).AND.(H2.GE.H3) ...
```

```
          .AND.(H2.LE.ABS((CI-CF)*.01))) TC=TIME
*  CHECK FOR THE LAST THREE OUTPUT VALUES TO BE DECREASING AND ALL LESS
*  THAN 1% OF THE DIFFERENCE BETWEEN THE INITIAL AND FINAL OUTPUT VALUES
          IF((H3.LT.H2).AND.(H2.LT.H1).AND.(H1.LT.  ...
           ABS((CI-CF)*.01))) TC=TIME
*  FOR OVER DAMPED CASE
*  CHECK FOR THE OUTPUT AT 5 TIMES THE TIME CONSTANT TO BE LESS
*  THAN 1% OF THE DIFFERENCE BETWEEN THE INITIAL AND FINAL OUTPUT VALUES
          IF((C.GT.(CF-ABS(CI-CF)*.01)).AND.(TIME.GE.(5.*TAU)) ...
          .AND.(TAU.NE.0.).AND.(C.EQ.CMAX)) TC=TIME
*  IF COMPLETION TIME NOT MET SET CONSTRAINT POSITIVE
          CON(2)=10.
*  SET CONSTRAINT VALUE NEGATIVE BY VALUE FROM TIME SPECIFICATION
          IF(TC.NE.0.) CON(2)=TC-TSP
*  MAX ALLOWABLE CONTROL VECTOR CONSTRAINT (CON(3))
*  (+) IF ABOVE SPECIFICATION  (-) IF BELOW SPECIFICATION
   150 IF(DELU.EQ.UMAX) CON(3)=DELU-USP
*  UNSTABLE CONSTRAINT CHECK (CON(4))
*  (+1) IF OUTPUT IS TEN TIMES THE MAX ALLOWABLE  (-1) OTHERWISE
          CON(4)=-1.
          IF(ABS(C).GT.ABS(CI-CF)*10.*(CSP/100.+1.))CON(4)=1.
*  TERMINATE RUN EARLY IF UNSTABLE TO PREVENT "BLOWING UP"
          IF(CON(4).GT.0.) CALL ENDRUN
*  NEGATIVE RESPONSE CONSTRAINT CHECK (CON(5))
*  (+1) IF OUTPUT INITIALLY GOES NEGATIVE  (-1) OTHERWISE
          CON(5)=-1.
          IF((ABS(CF-C).GT.ABS(CF-CI)).AND.(TAU.EQ.0.))CON(5)=1.
*  TERMINATE RUN EARLY IF UNSTABLE TO PREVENT "BLOWING UP"
          IF(CON(5).GT.0.) CALL ENDRUN
*  MAX ALLOWABLE X2 STATE CONSTRAINT (CON(6))
*  (+) IF ABOVE SPECIFICATION  (-) IF BELOW SPECIFICATION
          IF(DELX2.EQ.X2MAX) CON(6)=DELX2-X2SP
*  INTEGRATION COMPLETE CHECK
     30 IF((OBJ.LE.0.).OR.(LSTOBJ.LE.0.))GOTO 40
*  FOR OBJECTIVE FUNCTIONS THAT APPROACH A STEADY STATE VALUE CHECK FOR
```

```
* A OBJECTIVE FUNCTION SLOPE OF LESS THAN .1% OTHERWISE CHECK FOR THE
* SECOND DERIVATIVE OF THE OBJECTIVE FUNCTION TO BE LESS THAN .1%
* ALONG WITH COMPLETION TIME BEING REACHED IN EITHER CASE
         SLOPE=(OBJ-LSTOBJ)/(TIME-LSTTIM)
         DSLOPE=(SLOPE-LSLOPE)/(TIME-LSTTIM)
         IF(((SLOPE.LT..001).OR.(DSLOPE.LE..001))...
         .AND.(TC.NE.0.)) CALL ENDRUN
* ADVANCE THE LAST FUNCTION, TIME AND SLOPE COUNTERS FOR NEXT INTERVAL
    40 LSTOBJ=OBJ
         LSTTIM=TIME
         LSLOPE=SLOPE
*************************************************************************
TERMINAL
         IF (INFO.NE.0) GOTO 450
         DO 400 I=1,3
         DO 410 J=1,3
 410  V(I,J)=AA(I,J)-BB(I)*G(J)
 400  CONTINUE
         WRITE(6,499)G(1),G(2),G(3)
 499  FORMAT('0 FINAL GAINS G(1),G(2),G(3) = ',3E15.4)
* CALL TO FIND THE EIGNVALES WHICH ARE IN TURN THE FINAL SYSTEM POLES
         CALL RG(NM,NN,V,WR,WI,MATZ,Z,IV1,FV1,IERR)
         IF(IERR.NE.0)WRITE(6,495)IERR
 495  FORMAT('0 ERROR FROM EISPACK',I3,'-FINAL POLES MAYBE WRONG')
         WRITE(6,497)(WR(I),WI(I),I=1,3)
 497  FORMAT('0 FINAL POLES ARE : ',3('(',E11.4,',',E11.4,' J)  '))
         CALL ENDJOB
 450  CALL RERUN
END
STOP
```

APPENDIX E

PROGRAM LISTINGS FOR POLE TO GAIN APPROXIMATIONS

1. DETERMINANT METHOD

The following listing would replace the initialization and initial segments of the DSL program listed in Appendix D, for using the determinant evaluation method to find the initial gains corresponding to desired initial system poles. The DERIVATIVE, DYNAMIC, and TERMINAL sections of the state variable program would remain unchanged.

```
TITLE DC MOTOR INTEGRAL CONTROL SYSTEM-KUO EXAMPLE 8-7
************************************************************************
* INITIALIZE THE ADS PARAMETERS
FIXED NRA,NGT,NCOLA,NRWK,NRIWK,IGRAD,NDV,NCON,IDG,IC,IWK
FIXED ISTRAT,IOPT,IONED,IPRINT,INFO
ARRAY DV(7),VLB(7),VUB(7),CON(5),IDG(5),IC(5),DF(7),WK(1000),IWK(1000)
D     DIMENSION WA(40,40)
PARAM NRA=40,NCOLA=40,NRWK=1000,NRIWK=1000,NDV=3,NCON=1,IGRAD=0,INFO=0
* ESTABLISH TYPES OF CONSTRAINTS (ADS MANUAL TABLE 5)
TABLE IDG(1-5)=3*-1,2*2
* LOWER BOUND FOR DESIGN VARIABLES
TABLE VLB(1-6)=3*-1000.,3*0.
* UPPER BOUND FOR DESIGN VARIABLES
TABLE VUB(1-6)=3*1000.,3*0.
* ADS OPTIMIZATION METHOD SELECTION
PARAM ISTRAT=9, IOPT=5, IONED=7, IPRINT=3140
************************************************************************
* INITIALIZE THE MODEL PARAMETERS
* SET CONTROL PARAMETERS
FIXED I,J,K,II,IER1,IERR,NN,NM,IV1,MATZ,SEED
ARRAY BB(3),H(2),R(2),G(3),WR(3),WI(3),IV1(3),FV1(3)
D     DIMENSION AA(3,3),CC(2,2),Q(2,2),F(2,2),V(3,3),Z(3,3)
```

```
* SET INITIAL PARAMETERS FOR POLE TO GAIN OETERMINATION
TABLE G(1-3)=-.38,.6,-6.
0     INTEGER NP,JOB,IPVT(3),INF
0     COMPLEX*16 P(3),PA(3,3),PM(3,3),POET(2),PWK(3)
EXCLUOE NP,JOB,IPVT,INF,P,PA,PM,POET,PWK
* ZERO ALL MATRICES
0     OATA AA,F,CC,Q/21*0./
TABLE BB(1-3)=3*0.,R(1-2)=2*0.,H(1-2)=2*0.
* SET CONSTRAINT SPECIFICATIONS
* CI=INITIAL OUTPUT VALUE;  CF=FINAL EXPECTEO OUTPUT VALUE
* CSP=MAXIMUM PERCENT OVERSHOOT FOR THE OUTPUT
* TSP=MAXIMUM COMPLETION TIME
* USP=MAXIMUM VALUE OF THE CONTROL VECTOR
PARAM CI=1.,CF=0.,CSP=40.,TSP=1.5,USP=10.
* SET INITIAL INPUT VALUES (W1=OISTURBANCE, W2=REFERENCE SETTING)
PARAM W1=0., W2=1., SEEO=113, MEAN=1., SO=0.1, WL=1.6
* SET INITIAL RUN FLAG ANO EISPACK PARAMETERS
PARAM II=0,MATZ=0,NM=3,NN=3,I=1,J=1,K=1,IER1=0,IERR=0
*************************************************************************
* INITIALIZE THE OSL PARAMETERS
PRINT OBJ,C,X1,X2,U
SAVE (RUN) OBJ,X1,X2,U,C
CONTROL OELPLT=0.005,OELPRT=0.01
CONTROL FINTIM=4.
METHOD STIFF
RELERR OBJ=1.E-4
ABSERR OBJ=1.E-4
*************************************************************************
INITIAL
* CHECK FOR INITIAL RUN
         IF(II.GT.0) GOTO 200
* THE DESIGN VARIABLES ARE THE FEEOBACK GAINS
* INITIAL SYSTEM POLES TO BASE FEEDBACK GAINS UPON (COMPLEX FORM)
         P(1)=CMPLX(-10.0,10.0)
         P(2)=CMPLX(-10.0,-10.0)
```

```
        P(3)=CMPLX(-300.0,0.)
* SET NON-ZERO COEFFICIENTS
        AA(1,2)=50.
        AA(2,1)=-200.
        AA(2,2)=-200.
        AA(3,1)=-1.
        BB(2)=200.
        CC(1,1)=-1.
        H(2)=1.
        F(1,1)=-50.
        Q(1,1)=1.
        Q(2,2)=1.
        R(1)=1.
* SET THE DESIGN VARIABLE MATRIX
        DV(1)=G(1)
        DV(2)=G(2)
        DV(3)=G(3)
        DV(4)=Q(1,1)
        DV(5)=Q(2,2)
        DV(6)=R(1)
* ROUTINE TO FIND THE SYSTEM GAINS FROM KNOWN POLES USING ADS
        NP=3
        JOB=10
        INFO=0
* USE ADS TO FIND G'S THAT MAKE SUM(DET(RELATION MATRIX)) = 0.
  660 CALL ADS(INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,DV,VLB,...
        VUB,OBJ,CON,IDG,NGT,IC,DF,WA,NRA,NCOLA,WK,NRWK,IWK,NRIWK)
        IF(INFO.EQ.0)GOTO 650
        DET=0.
* CALCULATE THE RELATION MATRIX: (P-A+B*G)
*     DO 600 K=1,NP
        DO 610 I=1,NP
        DO 620 J=1,NP
  620 PA(I,J)=(-AA(I,J))
  610 PA(I,I)=P(I)-AA(I,I)
```

```
      DO 630 I=1,NP
      G(I)=DV(I)
      DO 640 J=1,NP
  640 PM(J,I)=PA(J,I)+BB(J)*G(I)
  630 CONTINUE
* CALL TO CONDITION THE RELATION MATRIX
      CALL ZGEFA(PM,NP,NP,IPVT,INF)
* CALL TO FIND DERTERMINATE OF RELATION MATRIX
      CALL ZGEDI(PM,NP,NP,IPVT,PDET,PWK,JOB)
* FIND OBJECTIVE FUNCTION-LOOKING FOR MINIMUM TO BE ZERO
      CON(K)=ABS(REAL(PDET(1)*10.**PDET(2)))
      OBJ=OBJ+CON(K)
  600 CONTINUE
      GOTO 660
* OUTPUT INITIAL POLE AND GAINS INFORMATION
  650 WRITE(6,297)P(1),P(2),P(3)
  297 FORMAT('0 DESIRED POLES WERE : ',3('(',E11.4,',',E11.4,' J)  '))
      WRITE(6,299)G(1),G(2),G(3)
  299 FORMAT('0 INITIAL GAINS G(1),G(2),G(3) = ',3E15.4)
* CALL TO FIND THE EIGNVALES WHICH ARE IN TURN THE INITIAL SYSTEM POLES
      DO 680 I=1,3
      DO 690 J=1,3
  690 V(I,J)=AA(I,J)-BB(I)*G(J)
  680 CONTINUE
      CALL RG(NM,NN,V,WR,WI,MATZ,Z,IV1,FV1,IERR)
      IF(IERR.NE.0)WRITE(6,485)IERR
  485 FORMAT('0 ERROR FROM EISPACK',I3,'-INITIAL POLES MAYBE WRONG')
      WRITE(6,487)(WR(I),WI(I),I=1,3)
  487 FORMAT('0 INITIAL POLES ARE : ',3('(',E11.4,',',E11.4,' J)  '))
      WRITE(6,290)
  290 FORMAT('0 STATE CONTROL MATRIX (B  AB) = ')
      WRITE(6,298) BB(1),AA(1,1)*BB(1)+AA(1,2)*BB(2)
      WRITE(6,298) BB(2),AA(2,1)*BB(1)+AA(2,2)*BB(2)
  298 FORMAT(2E15.4)
      WRITE(6,291)
```

```
  291 FORMAT('0 OUTPUT CONTROL MATRIX (CB  CAB) = ')
      WRITE(6,298) CC(1,1)*BB(1)+CC(1,2)*BB(2),CC(1,1)*(AA(1,1)* ...
      BB(1)+AA(1,2)*BB(2))+CC(1,2)*(AA(2,1)*BB(1)+AA(2,2)*BB(2))
      WRITE(6,292)
  292 FORMAT('0 OBSERVABILITY MATRIX (C*  A*C*) = ')
      WRITE(6,298) CC(1,1),AA(1,1)*CC(1,1)+AA(2,1)*CC(1,2)
      WRITE(6,298) CC(1,2),AA(1,2)*CC(1,1)+AA(2,2)*CC(1,2)
* RESET ADS PARAMETERS FOR MODEL OPTIMIZATION
      CALL ENDJOB
      NCON=5
      ISTRAT=0
      IOPT=4
      IONED=7
      IPRINT=1010
      IDG(1)=0
      IDG(2)=0
      IDG(3)=0
* CALL TO ADS TO GET THE DESIGN VARIABLE FOR THE SYSTEM
  200 CALL ADS(INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,DV,VLB,...
      VUB,OBJ,CON,IDG,NGT,IC,DF,WA,NRA,NCOLA,WK,NRWK,IWK,NRIWK)
      G(1)=DV(1)
      G(2)=DV(2)
      G(3)=DV(3)
      Q(1,1)=DV(4)
      Q(2,2)=DV(5)
      R(1)=DV(6)
* CHECK FOR TYPE OF RUN IN ORDER TO SAVE INITIAL AND FINAL RESULTS
      IF(II.NE.0)GOTO 235
GRAPH (INITIAL/RUN,RU=1,DE=TEK618 ) TIME,X1,X2,U
      GOTO 245
  235 IF(INFO.NE.0)GOTO 240
      DELPRT=0.01
      DELPLT=0.01
GRAPH (FINAL/RUN,RU=*,DE=TEK618 ) TIME,X1,X2,U
      GOTO 245
```

```
 240  DELPRT=0.0
        DELPLT=0.0
 245  II=1
* INITIALIZE CONSTRAINT CHECK VALUES
* LAST THREE VALUES OF THE OUTPUT
        YOSC1=CI
        YOSC2=CI
        YOSC3=CI
* MAXIMUM VALUE OF THE OUTPUT
        CMAX=0.
* THE TIME CONSTANT OF THE OUTPUT
        TAU=0.
* TIME WHEN COMPLETION TIME SPECIFICATION IS MET
        TC=0.
* INITIAL VALUE OF THE CONTROL VECTOR
        UI=0.
* MAXIMUM VALUE OF THE CONTROL
        UMAX=0.
* VALUE OF THE OBJECTIVE FUNCTION FOR THE LAST INTEGRATION INTERVAL
        LSTOBJ=0.
* SLOPE OF THE OBJECTIVE FUNCTION FOR THE LAST INTEGRATION INTERVAL
        LSLOPE=0.
* VALUE OF TIME AT END OF LAST INTEGRATION INTERVAL
        LSTTIM=0.
*************************************************************************
```

2. EIGENVALUE METHOD

The following program listing would be used if the the eigenvalue approximation method was desired for finding the gains corresponding to the desired initial system poles. The DERIVATIVE, DYNAMIC, and TERMINAL sections of the state variable program in Appendix D remain unchanged, and would follow the listing below.

```
TITLE DC MOTOR INTEGRAL CONTROL SYSTEM-NO NOISE-KUO EXAMPLE 8-7
*************************************************************************
```

```
* INITIALIZE THE ADS PARAMETERS
FIXED NRA,NGT,NCOLA,NRWK,NRIWK,IGRAD,NDV,NCON,IDG,IC,IWK
FIXED ISTRAT,IOPT,IONED,IPRINT,INFO
ARRAY DV(7),VLB(7),VUB(7),CON(5),IDG(5),IC(5),DF(7),WK(1000),IWK(1000)
D     DIMENSION WA(40,40)
PARAM NRA=40,NCOLA=40,NRWK=1000,NRIWK=1000,NDV=3,NCON=4,IGRAD=0,INFO=-2
* ESTABLISH TYPES OF CONSTRAINTS (ADS MANUAL TABLE 5)
TABLE IDG(1-5)=4*-1,2
* LOWER BOUND FOR DESIGN VARIABLES
TABLE VLB(1-6)=3*-1000.,3*0.
* UPPER BOUND FOR DESIGN VARIABLES
TABLE VUB(1-6)=3*1000.,3*0.
* ADS OPTIMIZATION METHOD SELECTION
PARAM ISTRAT=1, IOPT=3, IONED=3, IPRINT=1100
*************************************************************************
* INITIALIZE THE MODEL PARAMETERS
* SET CONTROL PARAMETERS
FIXED I,J,K,II,IER1,IERR,NN,NM,IV1,MATZ,SEED
ARRAY BB(3),H(2),R(2),G(3),WR(3),WI(3),IV1(3),FV1(3)
D     DIMENSION AA(3,3),CC(2,2),Q(2,2),F(2,2),V(3,3),Z(3,3)
D     COMPLEX*16 P(3)
EXCLUDE P
* SET INITIAL PARAMETERS FOR POLE TO GAIN DETERMINATION
TABLE G(1-3)=3*-1.
* ZERO ALL MATRICES
D     DATA AA,F,CC,Q/21*0./
TABLE BB(1-3)=3*0.,R(1-2)=2*0.,H(1-2)=2*0.
* SET CONSTRAINT SPECIFICATIONS
* CI=INITIAL OUTPUT VALUE;  CF=FINAL EXPECTED OUTPUT VALUE
* CSP=MAXIMUM PERCENT OVERSHOOT FOR THE OUTPUT
* TSP=MAXIMUM COMPLETION TIME
* USP=MAXIMUM VALUE OF THE CONTROL VECTOR
PARAM CI=1.,CF=0.,CSP=100.,TSP=1.9,USP=1.
* SET INITIAL INPUT VALUES (W1=DISTURBANCE, W2=REFERENCE SETTING)
PARAM W1=0., W2=1., SEED=113, MEAN=1., SD=0.1, WL=1.6
```

```
* SET INITIAL RUN FLAG AND EISPACK PARAMETERS
PARAM II=0,MATZ=0,NM=3,NN=3,I=1,J=1,K=1,IER1=0,IERR=0
*************************************************************************
* INITIALIZE THE DSL PARAMETERS
PRINT OBJ,C,X1,X2,U
SAVE (RUN) OBJ,X1,X2,U,C
CONTROL DELPLT=0.005,DELPRT=0.01
CONTROL FINTIM=4.
METHOD STIFF
RELERR OBJ=1.E-4
ABSERR OBJ=1.E-4
*************************************************************************
INITIAL
* CHECK FOR INITIAL RUN
        IF(II.GT.0) GOTO 200
* THE DESIGN VARIABLES ARE THE FEEDBACK GAINS
* INITIAL SYSTEM POLES TO BASE FEEDBACK GAINS UPON (COMPLEX FORM)
        P(1)=CMPLX(-100.0,10.0)
        P(2)=CMPLX(-100.0,-10.0)
        P(3)=CMPLX(-300.0,0.)
* SET NON-ZERO COEFFICIENTS
        AA(1,2)=50.
        AA(2,1)=-200.
        AA(2,2)=-200.
        AA(3,1)=-1.
        BB(2)=200.
        CC(1,1)=-1.
        H(2)=1.
        F(1,1)=-50.
        Q(1,1)=1.
        Q(2,2)=1.
        R(1)=1.
* SET THE DESIGN VARIABLE MATRIX
        DV(1)=G(1)
        DV(2)=G(2)
```

```
        DV(3)=G(3)
        DV(4)=Q(1,1)
        DV(5)=Q(2,2)
        DV(6)=R(1)
        CALL ADS(INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,DV,VLB,...
        VUB,OBJ,CON,IDG,NGT,IC,DF,WA,NRA,NCOLA,WK,NRWK,IWK,NRIWK)
        IWK(2)=D
* ROUTINE TO FIND THE SYSTEM GAINS FROM KNOWN POLES USING ADS
  66D CALL ADS(INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,DV,VLB,...
        VUB,OBJ,CON,IDG,NGT,IC,DF,WA,NRA,NCOLA,WK,NRWK,IWK,NRIWK)
        IF(INFO.EQ.D)GOTO 650
* CALCULATE THE RELATION MATRIX: (A-B*G)
        DO 7DD I=1,3
        G(I)=DV(I)
        DO 71D J=1,3
 71D  V(J,I)=AA(J,I)-BB(J)*G(I)
 7DD  CONTINUE
* CALL TO FIND THE RELATION MATRIX EIGENVALUES
        CALL RG(NM,NN,V,WR,WI,MATZ,Z,IV1,FV1,IERR)
        IF(IERR.NE.D)WRITE(6,791)IERR
 791  FORMAT('0 ERROR FROM EISPACK',I3,'-FINAL POLES MAYBE WRONG')
        IF(MAX(WR(1),WR(2),WR(3)).GT.0.)G(1)=-G(1)
        CON(1)=ABS(MAX(WR(1),WR(2),WR(3))-MAX(REAL(P(1)),...
        REAL(P(2)),REAL(P(3))))
        CON(2)=ABS(MIN(WR(1),WR(2),WR(3))-MIN(REAL(P(1)),...
        REAL(P(2)),REAL(P(3))))
        CON(3)=ABS(MAX(WI(1),WI(2),WI(3))-MAX(IMAG(P(1)),...
        IMAG(P(2)),IMAG(P(3))))
        CON(4)=ABS(MIN(WI(1),WI(2),WI(3))-MIN(IMAG(P(1)),...
        IMAG(P(2)),IMAG(P(3))))
        OBJ=CON(1)+CON(2)+CON(3)+CON(4)
        GOTO 660
* RESET ADS PARAMETERS FOR PERFOMRANCE INDEX OPTIMIZATION
  65D NCON=5
        IDG(4)=2
```

```
      WRITE(6,296)P(1),P(2),P(3)
  296 FORMAT('0 DESIRED POLES ARE : ',3('(',E11.4,',',E11.4,' J)  '))
      WRITE(6,297)WR(1),WI(1),WR(2),WI(2),WR(3),WI(3)
  297 FORMAT('0 INITIAL POLES ARE : ',3('(',E11.4,',',E11.4,' J)  '))
      WRITE(6,299)G(1),G(2),G(3)
  299 FORMAT('0 INITIAL GAINS G(1),G(2),G(3) = ',3E15.4)
      WRITE(6,290)
      CALL ENDJOB
  290 FORMAT('0 STATE CONTROL MATRIX (B  AB) = ')
      WRITE(6,298) BB(1),AA(1,1)*BB(1)+AA(1,2)*BB(2)
      WRITE(6,298) BB(2),AA(2,1)*BB(1)+AA(2,2)*BB(2)
  298 FORMAT(2E15.4)
      WRITE(6,291)
  291 FORMAT('0 OUTPUT CONTROL MATRIX (CB  CAB) = ')
      WRITE(6,298) CC(1,1)*BB(1)+CC(1,2)*BB(2),CC(1,1)*(AA(1,1)* ...
      BB(1)+AA(1,2)*BB(2))+CC(1,2)*(AA(2,1)*BB(1)+AA(2,2)*BB(2))
      WRITE(6,292)
  292 FORMAT('0 OBSERVABILITY MATRIX (C*  A*C*) = ')
      WRITE(6,298) CC(1,1),AA(1,1)*CC(1,1)+AA(2,1)*CC(1,2)
      WRITE(6,298) CC(1,2),AA(1,2)*CC(1,1)+AA(2,2)*CC(1,2)
* CALL TO ADS TO GET THE DESIGN VARIABLE FOR THE SYSTEM
  200 CALL ADS(INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,DV,VLB,...
      VUB,OBJ,CON,IDG,NGT,IC,DF,WA,NRA,NCOLA,WK,NRWK,IWK,NRIWK)
      G(1)=DV(1)
      G(2)=DV(2)
      G(3)=DV(3)
      Q(1,1)=DV(4)
      Q(2,2)=DV(5)
      R(1)=DV(6)
* CHECK FOR TYPE OF RUN IN ORDER TO SAVE INITIAL AND FINAL RESULTS
      IF(II.NE.0)GOTO 235
GRAPH (INITIAL/RUN,RU=1,DE=TEK618 ) TIME,X1,X2,U
      GOTO 245
  235 IF(INFO.NE.0)GOTO 240
      DELPRT=0.01
```

```
         DELPLT=0.01
GRAPH (FINAL/RUN,RU=*,DE=TEK618 ) TIME,X1,X2,U
         GOTO 245
  240  DELPRT=0.0
         DELPLT=0.0
  245  II=1
* INITIALIZE CONSTRAINT CHECK VALUES
* LAST THREE VALUES OF THE OUTPUT
         YOSC1=CI
         YOSC2=CI
         YOSC3=CI
* MAXIMUM VALUE OF THE OUTPUT
         CMAX=0.
* THE TIME CONSTANT OF THE OUTPUT
         TAU=0.
* TIME WHEN COMPLETION TIME SPECIFICATION IS MET
         TC=0.
* INITIAL VALUE OF THE CONTROL VECTOR
         UI=0.
* MAXIMUM VALUE OF THE CONTROL
         UMAX=0.
* VALUE OF THE OBJECTIVE FUNCTION FOR THE LAST INTEGRATION INTERVAL
         LSTOBJ=0.
* SLOPE OF THE OBJECTIVE FUNCTION FOR THE LAST INTEGRATION INTERVAL
         LSLOPE=0.
* VALUE OF TIME AT END OF LAST INTEGRATION INTERVAL
         LSTTIM=0.
*************************************************************************
```

APPENDIX F

CASE 2 RESULTS FOR SCALING

1. RESULTS FOR STRATEGY 057 WITH SCALING APPLIED

```
 DSL/VS ..... RELEASE 1 ..... MODIFICATION LEVEL 1 ..... 1985


                              >>>> DSL SIMULATION INPUT DATA <<<<
            TITLE DC MOTOR INTEGRAL CONTROL SYSTEM-NO NOISE-KUO EXAMPLE 8-7
            FIXED NRA,NGT,NCOLA,NRWK,NRIWK,IGRAD,NDV,NCON,IDG,IC,IWK
            FIXED ISTRAT,ICPT,IONED,IPRINT,INFO,IPVT
            PARAM NRA=40,NCOLA=40,NRWK=1000,NRIWK=1000,NDV=3,NCON=5,IGRAD=0,INFO=0
            TABLE IDG(1-5)=3*0.2*2
            TABLE VLB(1-6)=3*-1000.,3*0.
            TABLE VUB(1-6)=3*1000.,3*0.
            PARAM ISTRAT=0, IOPT=5, IONED=7, IPRINT=3050
            SAVE OBJ,X1,X2,U
            GRAPH (DE=SPRINT ) TIME,OBJ,X1,X2,U
            PRINT OBJ,C,X1,X2,U
            CONTROL FINTIM=4.
            METHOD STIFF
            RELERR OBJ=1.E-4
            ABSERR OBJ=1.E-4
            FIXED I,II,IERI,IERR,NN,NM,IV1,MATZ
            TABLE BB(I-3)=3*0.,R(I-2)=3*0.,H(1-2)=3*0.
            PARAM CI=1.,CF=0.,CSP=40.,TSP=1.5,USP=10.
            PARAM W1=0., W2=1.
            PARAM II=0,MATZ=0,NM=3,NN=3,I=I,J=I,IERI=0,IERR=0
            END
***  KLENGH =  4245, KPOINT =  1775, AVAILABLE SPACE LEFT IN COMMON/CURVAL/ =  2470 DOUBLE-WORDS  ***
***  STIFF   INTEGRATION METHOD USED ***
  INITIAL POLES ARE : (-0.1000E+02, 0.1000E+02 J)  (-0.1000E+02,-0.1000E+02 J)  (-0.3000E+03, 0.0000E+00 J)
  INITIAL GAINS G(1),G(2),G(3) =     -0.3800E+00     0.6000E+00    -0.6000E+01
  STATE CONTROL MATRIX (B  AB) =
       0.0000E+00     0.1000E+05
       0.2000E+03    -0.4000E+05
  OUTPUT CONTROL MATRIX (CB  CAB) =
       0.0000E+00    -0.1000E+05
  OBSERVABILITY MATRIX (C*  A*C*) =
      -0.1000E+01     0.0000E+00
       0.0000E+00    -0.5000E+02
```

```
  AAAAA    DDDDDD     SSSSSS
A     A   D     D   S
A     A   D     D   S
AAAAAAA   D     D    SSSSS
A     A   D     D         S
A     A   D     D         S
A     A   DDDDDD    SSSSSS
```

F O R T R A N P R O G R A M

F O R

A U T O M A T E D D E S I G N S Y N T H E S I S

V E R S I O N 1.10

CONTROL PARAMETERS

ISTRAT = 0 IOPT = 5 IONED = 7 IPRINT = 3050

IGRAD = 0 NDV = 3 NCON = 5

*** DSL OUTPUT LISTING, GROUP 1 . STARTING RUN 1 ***

DC MOTOR INTEGRAL CONTROL SYSTEM-NO NOISE-KUO EXAMPLE 8-7

TIME	OBJ	C	X1	X2	U
0.00000E+00	0.00000E+00	1.0000	0.00000E+00	0.00000E+00	0.00000E+00
1.00000E-02	1.94922E-02	0.99486	5.13754E-03	2.52547E-02	4.67107E-02
2.00000E-02	3.78729E-02	0.97439	2.56087E-02	5.58440E-02	9.52865E-02
3.00000E-02	5.47989E-02	0.94014	5.98551E-02	8.04192E-02	0.15103
4.00000E-02	7.00222E-02	0.89509	0.10491	9.91254E-02	0.21202
5.00000E-02	8.34120E-02	0.84193	0.15807	0.11263	0.27627
6.00000E-02	9.49395E-02	0.78319	0.21681	0.12165	0.34196
7.00000E-02	0.10467	0.72092	0.27908	0.12683	0.40765
8.00000E-02	0.11272	0.65689	0.34311	0.12878	0.47215
9.00000E-02	0.11926	0.59259	0.40741	0.12805	0.53451
1.00000E-01	0.12447	0.52921	0.47079	0.12513	0.59399
0.11000	0.12854	0.46775	0.53225	0.12047	0.65003
0.12000	0.13165	0.40896	0.59104	0.11448	0.70225
0.13000	0.13397	0.35343	0.64657	0.10751	0.75039
0.14000	0.13567	0.30157	0.69843	9.98488E-02	0.79433
0.15000	0.13688	0.25365	0.74635	9.17725E-02	0.83402
0.16000	0.13771	0.20983	0.79017	8.34994E-02	0.86952
0.17000	0.13827	0.17015	0.82985	7.52150E-02	0.90094
0.18000	0.13862	0.13459	0.86541	6.70709E-02	0.92847
0.19000	0.13884	0.10304	0.89696	5.91887E-02	0.95229
0.20000	0.13896	7.53384E-02	0.92466	5.16647E-02	0.97267
0.21000	0.13903	5.12988E-02	0.94870	4.45697E-02	0.98984
0.22000	0.13906	3.06886E-02	0.96931	3.79544E-02	1.0041
0.23000	0.13908	1.32590E-02	0.98674	3.18513E-02	1.0157
0.24000	0.13910	-1.25109E-03	1.0013	2.62778E-02	1.0249
0.25000	0.13911	-1.31081E-02	1.0131	2.12392E-02	1.0320
0.26000	0.13913	-2.25784E-02	1.0226	1.67296E-02	1.0372
0.27000	0.13915	-2.99229E-02	1.0299	1.27346E-02	1.0408
0.28000	0.13918	-3.53949E-02	1.0354	9.23358E-03	1.0430
0.29000	0.13922	-3.92343E-02	1.0392	6.20011E-03	1.0440
0.30000	0.13925	-4.16677E-02	1.0417	3.60431E-03	1.0441
0.31000	0.13929	-4.29059E-02	1.0429	1.41384E-03	1.0433
0.32000	0.13933	-4.31429E-02	1.0431	-4.05144E-04	1.0419
0.33000	0.13936	-4.25561E-02	1.0426	-1.88703E-03	1.0400
0.34000	0.13940	-4.13057E-02	1.0413	-3.06614E-03	1.0377
0.35000	0.13943	-3.95339E-02	1.0395	-3.97597E-03	1.0352
0.36000	0.13945	-3.73683E-02	1.0374	-4.64895E-03	1.0324
0.37000	0.13947	-3.49191E-02	1.0349	-5.11580E-03	1.0296
0.38000	0.13949	-3.22817E-02	1.0323	-5.40541E-03	1.0268
0.39000	0.13951	-2.95377E-02	1.0295	-5.54461E-03	1.0240
0.40000	0.13952	-2.67572E-02	1.0268	-5.55833E-03	1.0212
0.41000	0.13953	-2.39965E-02	1.0240	-5.46898E-03	1.0186
0.42000	0.13954	-2.13020E-02	1.0213	-5.29686E-03	1.0161
0.43000	0.13955	-1.87104E-02	1.0187	-5.06008E-03	1.0138
0.44000	0.13955	-1.62500E-02	1.0163	-4.77465E-03	1.0116

```
0.45000      0.13956     -1.39415E-02   1.0139     -4.45455E-03   1.0097
0.46000      0.13956     -1.17992E-02   1.0118     -4.11187E-03   1.0079
0.47000      0.13956     -9.83171E-03   1.0098     -3.75689E-03   1.0063


      SCALAR PRDGRAM PARAMETERS
      REAL PARAMETERS
          1) ALAMDZ =  0.00000E+00     20) EXTRAP =  0.50000E+01
          2) BETAMC =  0.00000E+00     21) FDCH   =  0.10000E-01
          3) CT     = -0.30000E-01     22) FDCHM  =  0.10000E-02
          4) CTL    = -0.50000E-02     23) GMULTZ =  0.10000E+02
          5) CTLMIN =  0.10000E-02     24) PSAIZ  =  0.95000E+00
          6) CTMIN  =  0.10000E-01     25) RMULT  =  0.50000E+01
          7) DABALP =  0.10000E-03     26) RMVLMZ =  0.20000E+00
          8) DABDBJ =  0.13956E-03     27) RP     =  0.10000E+02
          9) DABDSM =  0.27913E-03     28) RPMAX  =  0.10000E+11
         10) DABSTR =  0.13956E-03     29) RMULT  =  0.20000E+00
         11) DELALP =  0.50000E-02     30) RPPMIN =  0.10000E-09
         12) DELDBJ =  0.10000E-02     31) RPPRIM =  0.10000E+03
         13) DELDSM =  0.10000E-01     32) SCFD   =  0.10000E+01
         14) DELSTR =  0.10000E-02     33) SCLMIN =  0.10000E-02
         15) DLDBJ1 =  0.10000E+00     34) STDL   =  0.10000E-02
         16) DLDBJ2 =  0.10000E+04     35) THETAZ =  0.10000E+00
         17) DX1    =  0.10000E-01     36) XMULT  =  0.26180E+01
         18) DX2    =  0.20000E+00     37) ZRD    =  0.10000E-04
         19) EPSPEN = -0.50000E-01     38) PMLT   =  0.10000E+02

      INTEGER PARAMETERS
          1) ICNDIR =    4       4) ITRMDP =    3       6) JDNED  =    7
          2) ISCAL  =    1       5) ITRMST =    2       7) JTMAX  =   20
          3) ITMAX  =   40

      ARRAY STORAGE REQUIREMENTS
                    DIMENSIONED  REQUIRED
      ARRAY     SIZE         SIZE
          WK      1000          299
         IWK      1000          196


      SCALING INFORMATION CALCULATED BY ADS

      OBJECTIVE FUNCTION SCALE FACTOR, SCFD = 0.46685E+02

      DESIGN VARIABLE SCALE FACTORS
              1 0.65528E+01 0.16842E+01 0.37478E+00

      CONSTRAINT SCALE FACTORS

      CONSTRAINT NUMBERS
                  1           5           4           2           3

      SCALE FACTORS
        0.68265E+01 0.10000E+02 0.10000E+02 0.10147E+00 0.73710E+01



      ------------------------------------------------------------------------
                                 IOPT = 5: MODIFIED METHOD OF FEASIBLE DIRECTIONS
      ------------------------------------------------------------------------

      -- INITIAL DESIGN

      OBJ = 0.13956E+00

      DECISION VARIABLES (X-VECTOR)
            1)  -0.38000E+00    0.60000E+00  -0.60000E+01

      LOWER BOUNDS ON THE DECISION VARIABLES (VLB-VECTOR)
            1)  -0.10000E+04  -0.10000E+04  -0.10000E+04
```

```
UPPER BOUNDS ON THE DECISION VARIABLES (VUB-VECTOR)
      1)   0.10000E+04   0.10000E+04   0.10000E+04

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.24331E+01  -0.1036BE+00  -0.66014E+02  -0.10000E+02  -0.10000E+02


-- BEGIN ITERATION NUMBER    1

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (DF-VECTOR)
      1)   0.67675E+00   0.1262BE+01   0.22360E+01

SEARCH DIRECTION (S-VECTOR)
      1)  -0.30266E+00  -0.56474E+00  -0.10000E+01

PROPOSED ALPHA =  0.20000E+00

CALCULATED ALPHA =  0.32652E+01

OBJECTIVE = 0.57094E-01

DECISION VARIABLES (X-VECTOR)
      1)  -0.53081E+00  -0.4948BE+00  -0.14712E+02

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.14373E+01  -0.1244BE+00  -0.64494E+02  -0.10000E+02  -0.10000E+02


-- BEGIN ITERATION NUMBER    2

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (DF-VECTOR)
      1)  -0.49351E+00   0.53307E+00  -0.19154E-01

SEARCH DIRECTION (S-VECTOR)
      1)   0.70557E+00  -0.10000E+01  -0.23626E+00

PROPOSED ALPHA =  0.17326E+01

CALCULATED ALPHA =  0.25660E-01

OBJECTIVE = 0.56591E-01

DECISION VARIABLES (X-VECTOR)
      1)  -0.52B05E+00  -0.51012E+00  -0.14729E+02

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.14636E+01  -0.12500E+00  -0.64511E+02  -0.10000E+02  -0.10000E+02


-- BEGIN ITERATION NUMBER    3

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (DF-VECTOR)
      1)  -0.48796E+00   0.59645E+00  -0.22670E-01

SEARCH DIRECTION (S-VECTOR)
      1)   0.75708E+00  -0.10000E+01  -0.11070E+00
```

```
PROPOSED ALPHA =  0.16455E+01

CALCULATED ALPHA =  0.69B95E-02

OBJECTIVE = 0.56456E-01

DECISION VARIABLES (X-VECTOR)
      1)  -0.52724E+00  -0.51427E+00  -0.14731E+02

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.14745E+01  -0.12506E+00  -0.64519E+02  -0.10000E+02  -0.10000E+02


-- BEGIN ITERATION NUMBER    4

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (OF-VECTOR)
      1)  -0.48500E+00   0.52880E+00  -0.23835E-01

SEARCH DIRECTION (S-VECTOR)
      1)   0.90812E+00  -0.10000E+01  -0.61035E-01

PROPOSED ALPHA =  0.16325E-01

CALCULATED ALPHA =  0.22228E+00

OBJECTIVE = 0.53864E-01

DECISION VARIABLES (X-VECTOR)
      1)  -0.49983E+00  -0.64625E+00  -0.14767E+02

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.17541E+01  -0.12181E+00  -0.64550E+02  -0.10000E+02  -0.10000E+02


-- BEGIN ITERATION NUMBER    5

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (OF-VECTOR)
      1)  -0.47B29E+00  -0.33528E+00  -0.87296E-01

SEARCH DIRECTION (S-VECTOR)
      1)   0.10000E+01   0.70100E+00   0.1B252E+00

PROPOSED ALPHA =  0.11464E+00

CALCULATED ALPHA =  0.17350E+00

OBJECTIVE = 0.52701E-01

DECISION VARIABLES (X-VECTOR)
      1)  -0.47335E+00  -0.57404E+00  -0.146B2E+02

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.20409E+01  -0.12899E+00  -0.65109E+02  -0.10000E+02  -0.10000E+02


-- BEGIN ITERATION NUMBER    6

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS
```

```
GRADIENT OF THE OBJECTIVE FUNCTION (DF-VECTOR)
      1)  -0.24605E+00   0.38073E+00   0.44972E-01

SEARCH DIRECTION (S-VECTOR)
      1)   0.10000E+01  -0.34160E+00   0.13126E-01

PROPOSED ALPHA =  0.19789E+00

CALCULATED ALPHA =  0.54227E+00

OBJECTIVE = 0.50417E-01

DECISION VARIABLES (X-VECTOR)
      1)  -0.39060E+00  -0.68402E+00  -0.14663E+02

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.26250E+01  -0.12582E+00  -0.65743E+02  -0.10000E+02  -0.10000E+02


-- BEGIN ITERATION NUMBER    7

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (DF-VECTOR)
      1)  -0.52639E-01  -0.95469E-01   0.10113E+00

SEARCH DIRECTION (S-VECTOR)
      1)   0.10000E+01   0.69751E+00  -0.91949E+00

PROPOSED ALPHA =  0.35788E+00

CALCULATED ALPHA =  0.99532E-01

OBJECTIVE = 0.50195E-01

DECISION VARIABLES (X-VECTOR)
      1)  -0.37541E+00  -0.64280E+00  -0.14908E+02

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.26885E+01  -0.13443E+00  -0.65935E+02  -0.10000E+02  -0.10000E+02


-- BEGIN ITERATION NUMBER    8

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (DF-VECTOR)
      1)  -0.19434E-02   0.22563E+00   0.14040E+00

SEARCH DIRECTION (S-VECTOR)
      1)   0.76057E+00   0.38162E-01  -0.10000E+01

PROPOSED ALPHA =  0.32090E+00

CALCULATED ALPHA =  0.27000E+03

OBJECTIVE = 0.35264E-01

DECISION VARIABLES (X-VECTOR)
      1)   0.30963E+02   0.54750E+01  -0.73533E+03

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.27515E+01  -0.12889E+00  -0.66352E+02  -0.10000E+02  -0.10000E+02
```

```
-- BEGIN ITERATION NUMBER    9

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (OF-VECTOR)
      1)   0.70317E-02   0.13976E-01   0.59860E-02

SEARCH DIRECTION (S-VECTOR)
      1)  -0.50312E+00  -0.10000E+01  -0.42831E+00

PROPOSED ALPHA =  0.13505E+03

CALCULATED ALPHA =  0.00000E+00

OBJECTIVE = 0.35264E-01

DECISION VARIABLES (X-VECTOR)
      1)   0.30963E+02   0.54750E+01  -0.73533E+03

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.27515E+01  -0.12889E+00  -0.66352E+02  -0.10000E+02  -0.10000E+02


-- BEGIN ITERATION NUMBER   10

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (OF-VECTOR)
      1)   0.70317E-02   0.13976E-01   0.59860E-02

SEARCH DIRECTION (S-VECTOR)
      1)  -0.50312E+00  -0.10000E+01  -0.42831E+00

PROPOSED ALPHA =  0.13500E+03

CALCULATED ALPHA =  0.00000E+00

OBJECTIVE = 0.35264E-01

DECISION VARIABLES (X-VECTOR)
      1)   0.30963E+02   0.54750E+01  -0.73533E+03

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.27515E+01  -0.12889E+00  -0.66352E+02  -0.10000E+02  -0.10000E+02


-- BEGIN ITERATION NUMBER   11

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (OF-VECTOR)
      1)   0.70317E-02   0.13976E-01   0.59860E-02

SEARCH DIRECTION (S-VECTOR)
      1)  -0.50312E+00  -0.10000E+01  -0.42831E+00

PROPOSED ALPHA =  0.10000E-02

CALCULATED ALPHA =  0.57764E+01

OBJECTIVE = 0.33286E-01
```

```
DECISION VARIABLES (X-VECTOR)
      1)   0.30520E+02   0.20453E+01  -0.74193E+03

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.27600E+01  -0.12966E+00  -0.66357E+02  -0.10000E+02  -0.10000E+02


-- BEGIN ITERATION NUMBER   12

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (DF-VECTOR)
      1)   0.72741E-02   0.68029E-01   0.56003E-02

SEARCH DIRECTION (S-VECTOR)
      1)  -0.45297E+00  -0.10000E+01  -0.38451E+00

PROPOSED ALPHA =  0.28882E+01

CALCULATED ALPHA =  0.50260E+00

OBJECTIVE = 0.33279E-01

DECISION VARIABLES (X-VECTOR)
      1)   0.30485E+02   0.17469E+01  -0.74245E+03

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.27534E+01  -0.12857E+00  -0.66285E+02  -0.10000E+02  -0.10000E+02


-- BEGIN ITERATION NUMBER   13

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (DF-VECTOR)
      1)   0.70757E-02   0.44616E-02   0.54342E-02

SEARCH DIRECTION (S-VECTOR)
      1)  -0.10000E+01  -0.63056E+00  -0.76801E+00

PROPOSED ALPHA =  0.31395E+01

CALCULATED ALPHA =  0.10690E+01

OBJECTIVE = 0.33101E-01

DECISION VARIABLES (X-VECTOR)
      1)   0.30322E+02   0.13466E+01  -0.74464E+03

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.27598E+01  -0.12961E+00  -0.65688E+02  -0.10000E+02  -0.10000E+02


-- BEGIN ITERATION NUMBER   14

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (DF-VECTOR)
      1)   0.72210E-02  -0.10101E-01   0.59956E-02

SEARCH DIRECTION (S-VECTOR)
      1)  -0.10000E+01   0.75990E-01  -0.78970E+00
```

```
PROPOSED ALPHA =  0.7B5B1E+00

CALCULATED ALPHA =  0.58461E+02

OBJECTIVE = 0.24306E-01

DECISION VARIABLES (X-VECTOR)
      1)   0.21400E+02   0.39843E+01  -0.B67B2E+03

CONSTRAINT VALUES (G-VECTOR)
      1) -0.27575E+01  -0.14014E+00  -0.66347E+02  -0.10000E+02  -0.10000E+02


-- BEGIN ITERATION NUMBER   15

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (DF-VECTOR)
      1)   0.22096E-02   0.10575E-01   0.14623E-02

SEARCH DIRECTION (S-VECTOR)
      1) -0.10000E+01  -0.63193E+00  -0.77107E+00

PROPOSED ALPHA =  0.29765E+02

CALCULATED ALPHA =  0.00000E+00

OBJECTIVE = 0.24306E-01

DECISION VARIABLES (X-VECTOR)
      1)   0.21400E+02   0.39B43E+01  -0.86782E+03

CONSTRAINT VALUES (G-VECTOR)
      1) -0.27575E+01  -0.14014E+00  -0.66347E+02  -0.10000E+02  -0.10000E+02


-- BEGIN ITERATION NUMBER   16

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (DF-VECTOR)
      1)   0.22096E-02   0.10575E-01   0.14623E-02

SEARCH DIRECTION (S-VECTOR)
      1) -0.20B95E+00  -0.10000E+01  -0.13B28E+00

PROPOSED ALPHA =  0.29230E+02

CALCULATED ALPHA =  0.00000E+00

OBJECTIVE = 0.24306E-01

DECISION VARIABLES (X-VECTOR)
      1)   0.21400E+02   0.39843E+01  -0.86782E+03

CONSTRAINT VALUES (G-VECTOR)
      1) -0.27575E+01  -0.14014E+00  -0.66347E+02  -0.10000E+02  -0.10000E+02


-- BEGIN ITERATION NUMBER   17

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS
```

```
GRAOIENT OF THE O8JECTIVE FUNCTION (OF-VECTOR)
      I)   0.22096E-02   0.I0575E-0I   0.14623E-02

SEARCH DIRECTION (S-VECTOR)
      I)  -0.20895E+00  -0.I0000E+0I  -0.13828E+00

PROPOSED ALPHA =  0.10000E-02

CALCULATED ALPHA =  0.22054E+01

OBJECTIVE = 0.239I4E-0I

OECISION VARIA8LES (X-VECTOR)
      I)   0.2I330E+02   0.26749E+0I  -0.86863E+03

CONSTRAINT VALUES (G-VECTOR)
      I)  -0.27529E+0I  -0.I390IE+00  -0.6524IE+02  -0.I0000E+02  -0.I0000E+02


-- BEGIN ITERATION NUMBER   I8

THERE ARE    0 ACTIVE CONSTRAINTS ANO    0 VIOLATEO CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRAOIENT OF THE O8JECTIVE FUNCTION (OF-VECTOR)
      I)   0.29993E-02   0.48979E-02   0.1I110E-02

SEARCH OIRECTION (S-VECTOR)
      I)  -0.38877E+00  -0.I0000E+0I  -0.I7775E+00

PROPOSEO ALPHA =  0.1I027E+0I

CALCULATEO ALPHA =  0.95684E-0I

O8JECTIVE = 0.239I3E-0I

OECISION VARIA8LES (X-VECTOR)
      I)   0.2I324E+02   0.26I80E+0I  -0.86868E+03

CONSTRAINT VALUES (G-VECTOR)
      I)  -0.27632E+0I  -0.I3983E+00  -0.65I54E+02  -0.I0000E+02  -0.I0000E+02


-- BEGIN ITERATION NUMBER   I9

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATEO CONSTRAINTS

THERE ARE    0 ACTIVE SIOE CONSTRAINTS

GRAOIENT OF THE O8JECTIVE FUNCTION (OF-VECTOR)
      I)   0.259I5E-02   0.25879E-0I   0.I3I33E-02

SEARCH OIRECTION (S-VECTOR)
      I)  -0.I00I4E+00  -0.I0000E+0I  -0.50749E-0I

PROPOSEO ALPHA =  0.11505E+0I

CALCULATEO ALPHA =  0.00000E+00

O8JECTIVE = 0.23913E-0I

DECISION VARIABLES (X-VECTOR)
      I)   0.2I324E+02   0.26I80E+0I  -0.86868E+03

CONSTRAINT VALUES (G-VECTOR)
      I)  -0.27632E+0I  -0.I3983E+00  -0.65I54E+02  -0.I0000E+02  -0.I0000E+02
```

```
-- BEGIN ITERATION NUMBER   20

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (DF-VECTOR)
      1)   0.25915E-02   0.25879E-01   0.13133E-02

SEARCH DIRECTION (S-VECTOR)
      1) -0.10014E+00  -0.10000E+01  -0.50749E-01

PROPOSED ALPHA =  0.47842E-01

CALCULATED ALPHA =  0.97728E-02

OBJECTIVE = 0.23913E-01

DECISION VARIABLES (X-VECTOR)
      1)   0.21324E+02   0.26122E+01  -0.86868E+03

CONSTRAINT VALUES (G-VECTOR)
      1) -0.27635E+01  -0.13985E+00  -0.65146E+02  -0.10000E+02  -0.10000E+02



FINAL OPTIMIZATION RESULTS

NUMBER OF ITERATIONS =   20

OBJECTIVE = 0.2391?E-01

DECISION VARIABLES (X-VECTOR)
      1)   0.21324E+02   0.26122E+01  -0.86868E+03

CONSTRAINT VALUES (G-VECTOR)
      1) -0.27635E+01  -0.13985E+00  -0.65146E+02  -0.10000E+02  -0.10000E+02

CONSTRAINT TOLERANCE, CT = -0.30000E-01  CTL = -0.50000E-02

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

TERMINATION CRITERIA

RELATIVE CONVERGENCE CRITERION WAS MET FOR  3 CONSECUTIVE ITERATIONS

ABSOLUTE CONVERGENCE CRITERION WAS MET FOR  3 CONSECUTIVE ITERATIONS

MAXIMUM K-T RESIDUAL =  0.00000E+00 IS LESS THAN  0.10000E-02


                    ----------------------
                      OPTIMIZATION RESULTS
                    ----------------------


OBJECTIVE FUNCTION VALUE    0.23913E-01


DESIGN VARIABLES

                          LOWER                          UPPER
VARIABLE     BOUND          VALUE          BOUND
        1   -0.10000E+04   0.21324E+02   0.10000E+04
        2   -0.10000E+04   0.26122E+01   0.10000E+04
        3   -0.10000E+04  -0.86868E+03   0.10000E+04
```

```
        DESIGN CONSTRAINTS

                1) -0.4048E+00 -0.1378E+01 -0.8838E+01 -0.1000E+01 -0.1000E+01

        FUNCTION EVALUATIONS =   173
                *** DSL OUTPUT LISTING, GROUP  1
                                                                        , STARTING RUN 175 ***
      TIME          O8J           C           XI           X2           U
  0.00000E+00  0.00000E+00   1.0000      0.00000E+00  0.00000E+00  0.00000E+00
  1.00000E-02  1.19744E-02  0.75301      0.24699      0.69497      0.82111
  2.00000E-02  1.91402E-02  0.47626      0.52374      0.42929      0.85409
  3.00000E-02  2.19849E-02  0.30355      0.69645      0.27368      0.90859
  4.00000E-02  2.31383E-02  0.19342      0.80658      0.17440      0.94170
  5.00000E-02  2.36031E-02  0.12329      0.87671      0.11115      0.96280
  6.00000E-02  2.37926E-02  7.85797E-02  0.92142      7.08448E-02  0.97630
  7.00000E-02  2.38697E-02  5.00829E-02  0.94992      4.51536E-02  0.98489
  8.00000E-02  2.39009E-02  3.19248E-02  0.96808      2.87808E-02  0.99037
  9.00000E-02  2.39136E-02  2.03445E-02  0.97966      1.83416E-02  0.99386
  1.00000E-01  2.39188E-02  1.29620E-02  0.98704      1.16859E-02  0.99609
  0.11000      2.39209E-02  8.25829E-03  0.99174      7.44523E-03  0.99751
  FINAL GAINS G(1),G(2),G(3) =      0.2132E+02     0.2612E+01    -0.8687E+03
  FINAL POLES ARE : (-0.3387E+03, 0.2793E+03 J)  (-0.3387E+03,-0.2793E+03 J)  (-0.4508E+02, 0.0000E+00 J)
SUMMARY:  ENDING RUN NO. =  175,  NO. OF RERUNS REQUESTED =  175
                *** RUN TERMINATED BY 'ENDJOB' ***
```

2. RESULTS FOR STRATEGY 057 WITHOUT SCALING APPLIED

```
        DSL/VS ..... RELEASE 1 ..... MODIFICATION LEVEL 1 ..... 1985


                                >>>> DSL SIMULATION INPUT DATA <<<<
            TITLE DC MOTOR INTEGRAL CONTROL SYSTEM-NO NOISE-KUO EXAMPLE 8-7
            FIXED NRA,NGT,NCOLA,NRWK,NRIWK,IGRAD,NDV,NCON,IDG,IC,IWK
            FIXED ISTRAT,IOPT,IONED,IPRINT,INFO,IPVT
            PARAM NRA=40,NCOLA=40,NRWK=1000,NRIWK=1000,NDV=3,NCON=5,IGRAD=0,INFO=-2
            TABLE IDG(1-5)=3*0,2*2
            TABLE VL8(1-6)=3*-1000.,3*0.
            TABLE VU8(1-6)=3*1000.,3*0.
            PARAM ISTRAT=0, IOPT=5, IONED=7, IPRINT=1050
            SAVE O8J,X1,X2,U
            GRAPH (DE=SPRINT ) TIME,O8J,X1,X2,U
            PRINT O8J,C,XI,X2,U
            CONTROL FINTIM=4.
            METHOD STIFF
            RELERR O8J=1.E-4
            A8SERR O8J=1.E-4
            FIXED I,II,IER1,IERR,NN,NM,IV1,MATZ
            TABLE 88(1-3)=3*0.,R(I-2)=3*0.,H(1-2)=3*0.
            PARAM CI=1.,CF=0.,CSP=40.,TSP=1.5,USP=10.
            PARAM W1=0., W2=I.
            PARAM II=0,MATZ=0,NM=3,NN=3,I=1,J=1,IER1=0,IERR=0
            END
***  KLENGH =  4245, KPOINT =  1775, AVAILABLE SPACE LEFT IN COMMON/CURVAL/ =  2470 DOUBLE-WORDS  ***
***  STIFF   INTEGRATION METHOD USED ***
  INITIAL POLES ARE : (-0.1000E+02, 0.1000E+02 J)  (-0.1000E+02,-0.1000E+02 J)  (-0.3000E+03, 0.0000E+00 J)
  INITIAL GAINS G(1),G(2),G(3) =     -0.3800E+00     0.6000E+00    -0.6000E+01
  STATE CONTROL MATRIX (8  A8) =
       0.0000E+00     0.1000E+05
```

```
      0.2000E+03    -0.4000E+05
OUTPUT CONTROL MATRIX (C8  CA8) =
      0.0000E+00    -0.1000E+05
OBSERVABILITY MATRIX (C*  A*C*) =
    -0.1000E+01     0.0000E+00
      0.0000E+00    -0.5000E+02
```

```
  AAAAA    DDDDDD     SSSSSS
A     A   D     D   S
A     A   D     D   S
AAAAAAA   D     D    SSSSS
A     A   D     D         S
A     A   D     D         S
A     A   DDDDDD    SSSSSS
```

FORTRAN PROGRAM

FDR

AUTOMATED DESIGN SYNTHESIS

VERSION 1.10

```
      CONTROL PARAMETERS
      ISTRAT =    0    IOPT   =    5    IONED  =    7    IPRINT = 1050
      IGRAD  =    0    NDV    =    3    NCON   =    5
          SCALING NOT USED
              *** DSL OUTPUT LISTING, GROUP  1                    , STARTING RUN   1 ***

                DC MOTOR INTEGRAL CONTROL SYSTEM-NO NOISE-KUO EXAMPLE 8-7
    TIME          OBJ           C           X1           X2           U
0.00000E+00  0.00000E+00   1.0000      0.00000E+00  0.00000E+00  0.00000E+00
1.00000E-02  1.94922E-02  0.99486      5.13754E-03  2.52547E-02  4.67107E-02
2.00000E-02  3.78729E-02  0.97439      2.56087E-02  5.58440E-02  9.52865E-02
3.00000E-02  5.47989E-02  0.94014      5.98551E-02  8.04192E-02  0.15103
4.00000E-02  7.00222E-02  0.89509      0.10491      9.91254E-02  0.21202
5.00000E-02  8.34120E-02  0.84193      0.15807      0.11263      0.27627
6.00000E-02  9.49395E-02  0.78319      0.21681      0.12165      0.34196
7.00000E-02  0.10467      0.72092      0.27908      0.12683      0.40765
8.00000E-02  0.11272      0.65689      0.34311      0.12878      0.47215
9.00000E-02  0.11926      0.59259      0.40741      0.12805      0.53451
1.00000E-01  0.12447      0.52921      0.47079      0.12513      0.59399
0.11000      0.12854      0.46775      0.53225      0.12047      0.65003
0.12000      0.13165      0.40896      0.59104      0.11448      0.70225
0.13000      0.13397      0.35343      0.64657      0.10751      0.75039
0.14000      0.13567      0.30157      0.69843      9.98488E-02  0.79433
0.15000      0.13688      0.25365      0.74635      9.17725E-02  0.83402
0.16000      0.13771      0.20983      0.79017      8.34994E-02  0.86952
0.17000      0.13827      0.17015      0.82985      7.52150E-02  0.90094
0.18000      0.13862      0.13459      0.86541      6.70709E-02  0.92847
0.19000      0.13884      0.10304      0.89696      5.91887E-02  0.95229
```

```
0.20000      0.13896      7.53384E-02  0.92466      5.16647E-02  0.97267
0.21000      0.13903      5.12988E-02  0.94870      4.45697E-02  0.98984
0.22000      0.13906      3.06886E-02  0.96931      3.79544E-02   1.0041
0.23000      0.13908      1.32590E-02  0.98674      3.13513E-02   1.0157
0.24000      0.13910     -1.25109E-03   1.0013      2.62778E-02   1.0249
0.25000      0.13911     -1.31081E-02   1.0131      2.12392E-02   1.0320
0.26000      0.13913     -2.25784E-02   1.0226      1.67296E-02   1.0372
0.27000      0.13915     -2.99229E-02   1.0299      1.27346E-02   1.0408
0.28000      0.13918     -3.53949E-02   1.0354      9.23358E-03   1.0430
0.29000      0.13922     -3.92343E-02   1.0392      6.20011E-03   1.0440
0.30000      0.13925     -4.16677E-02   1.0417      3.60431E-03   1.0441
0.31000      0.13929     -4.29059E-02   1.0429      1.41384E-03   1.0433
0.32000      0.13933     -4.31429E-02   1.0431     -4.05144E-04   1.0419
0.33000      0.13936     -4.25561E-02   1.0426     -1.88703E-03   1.0400
0.34000      0.13940     -4.13057E-02   1.0413     -3.06614E-03   1.0377
0.35000      0.13943     -3.95339E-02   1.0395     -3.97597E-03   1.0352
0.36000      0.13945     -3.73683E-02   1.0374     -4.64895E-03   1.0324
0.37000      0.13947     -3.49191E-02   1.0349     -5.11580E-03   1.0296
0.38000      0.13949     -3.22817E-02   1.0323     -5.40541E-03   1.0268
0.39000      0.13951     -2.95377E-02   1.0295     -5.54461E-03   1.0240
0.40000      0.13952     -2.67572E-02   1.0268     -5.55833E-03   1.0212
0.41000      0.13953     -2.39965E-02   1.0240     -5.46898E-03   1.0186
0.42000      0.13954     -2.13020E-02   1.0213     -5.29686E-03   1.0161
0.43000      0.13955     -1.87104E-02   1.0187     -5.06008E-03   1.0138
0.44000      0.13955     -1.62500E-02   1.0163     -4.77465E-03   1.0116
0.45000      0.13956     -1.39415E-02   1.0139     -4.45455E-03   1.0097
0.46000      0.13956     -1.17992E-02   1.0118     -4.11187E-03   1.0079
0.47000      0.13956     -9.83171E-03   1.0098     -3.75689E-03   1.0063
```

```
-------------------------------------------------------------------------
                          IOPT = 5: MODIFIED METHOD OF FEASIBLE DIRECTIONS
-------------------------------------------------------------------------

-- INITIAL DESIGN

OBJ = 0.13956E+00

DECISION VARIABLES (X-VECTOR)
      1)  -0.38000E+00   0.60000E+00  -0.60000E+01

LOWER BOUNDS ON THE DECISION VARIABLES (VLB-VECTOR)
      1)  -0.10000E+04  -0.10000E+04  -0.10000E+04

UPPER BOUNDS ON THE DECISION VARIABLES (VUB-VECTOR)
      1)   0.10000E+04   0.10000E+04   0.10000E+04

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.35685E+00  -0.10217E+01  -0.89558E+01  -0.10000E+01  -0.10000E+01


-- BEGIN ITERATION NUMBER    1

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (DF-VECTOR)
      1)   0.94990E-01   0.45556E-01   0.17951E-01

SEARCH DIRECTION (S-VECTOR)
      1)  -0.10000E+01  -0.47959E+00  -0.18897E+00

PROPOSED ALPHA =  0.11608E+00

CALCULATED ALPHA =  0.22788E+00

OBJECTIVE = 0.12996E+00
```

```
DECISION VARIABLES (X-VECTOR)
      1)  -0.607B8E+00   0.49071E+00  -0.60431E+01

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.19485E+00  -0.72B29E+00  -0.87893E+01  -0.10000E+01  -0.10000E+01


-- BEGIN ITERATION NUMBER    2

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (DF-VECTOR)
      1)  -0.59658E-01   0.81344E-01   0.11749E-01

SEARCH DIRECTION (S-VECTOR)
      1)  -0.21330E+00  -0.10000E+01  -0.22827E+00

PROPOSED ALPHA =  0.17198E+00

CALCULATED ALPHA =  0.90049E+00

OBJECTIVE = 0.10507E+00

DECISION VARIABLES (X-VECTOR)
      1)  -0.79995E+00  -0.40977E+00  -0.62486E+01

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.69459E-02  -0.78792E+00  -0.85804E+01  -0.10000E+01  -0.10000E+01


-- BEGIN ITERATION NUMBER    3

THERE ARE    1 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS
CONSTRAINT NUMBERS
        1

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (DF-VECTOR)
      1)  -0.43352E+00   0.78925E-01  -0.22729E-02

GRADIENT OF CONSTRAINT NUMBER    1
      1)  -0.24212E+01   0.16891E+00  -0.53977E-01

K-T PARAMETERS, BETA =  0.50816E+00  MAX. RESIDUAL =  0.491B4E+00

SEARCH DIRECTION (S-VECTOR)
      1)   0.10000E+01  -0.18206E+00   0.52429E-02

PROPOSED ALPHA =  0.56418E+00

CALCULATED ALPHA =  0.15068E+00

OBJECTIVE = 0.80689E-01

DECISION VARIABLES (X-VECTOR)
      1)  -0.64927E+00  -0.43721E+00  -0.6247BE+01

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.32129E+00  -0.11902E+01  -0.B9122E+01  -0.10000E+01  -0.10000E+01


-- BEGIN ITERATION NUMBER    4

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS
```

```
THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (DF-VECTOR)
      1)   0.12485E-01   0.52862E-01   0.67057E-02

SEARCH DIRECTION (S-VECTOR)
      1)  -0.23618E+00  -0.10000E+01  -0.12685E+00

PROPOSED ALPHA =  0.32559E+00

CALCULATED ALPHA =  0.52268E-02

OBJECTIVE = 0.80397E-01

DECISION VARIABLES (X-VECTOR)
      1)  -0.65050E+00  -0.44243E+00  -0.62485E+01

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.32116E+00  -0.11920E+01  -0.89119E+01  -0.10000E+01  -0.10000E+01


-- BEGIN ITERATION NUMBER    5

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (DF-VECTOR)
      1)   0.11906E-01   0.52925E-01   0.66478E-02

SEARCH DIRECTION (S-VECTOR)
      1)  -0.23055E+00  -0.10000E+01  -0.12623E+00

PROPOSED ALPHA =  0.77955E-01

CALCULATED ALPHA =  0.20409E+00

OBJECTIVE = 0.70223E-01

DECISION VARIABLES (X-VECTOR)
      1)  -0.69756E+00  -0.64652E+00  -0.62743E+01

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.30233E+00  -0.12122E+01  -0.88751E+01  -0.10000E+01  -0.10000E+01


-- BEGIN ITERATION NUMBER    6

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (DF-VECTOR)
      1)  -0.31121E-01   0.44775E-01   0.35499E-02

SEARCH DIRECTION (S-VECTOR)
      1)   0.45047E-01  -0.10000E+01  -0.11225E+00

PROPOSED ALPHA =  0.10466E+00

CALCULATED ALPHA =  0.10466E+00

OBJECTIVE = 0.67007E-01

DECISION VARIABLES (X-VECTOR)
      1)  -0.69284E+00  -0.75118E+00  -0.62860E+01

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.32042E+00  -0.11341E+01  -0.88657E+01  -0.10000E+01  -0.10000E+01
```

```
-- BEGIN ITERATION NUMBER    7

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (DF-VECTOR)
      1)  -0.44291E-01   0.57436E-02   0.18823E-02

SEARCH DIRECTION (S-VECTOR)
      1)   0.45896E+00  -0.10000E+01  -0.12388E+00

PROPOSED ALPHA =  0.15437E+00

CALCULATED ALPHA =  0.20074E-01

OBJECTIVE = 0.66636E-01

DECISION VARIABLES (X-VECTOR)
      1)  -0.68363E+00  -0.77125E+00  -0.62885E+01

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.34051E+00  -0.11226E+01  -0.88804E+01  -0.10000E+01  -0.10000E+01


-- BEGIN ITERATION NUMBER    8

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (DF-VECTOR)
      1)  -0.36555E-01  -0.67581E-02   0.19196E-02

SEARCH DIRECTION (S-VECTOR)
      1)   0.10000E+01   0.18488E+00  -0.52513E-01

PROPOSED ALPHA =  0.62366E-01

CALCULATED ALPHA =  0.24117E-01

OBJECTIVE = 0.66174E-01

DECISION VARIABLES (X-VECTOR)
      1)  -0.65951E+00  -0.76680E+00  -0.62898E+01

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.37897E+00  -0.11891E+01  -0.89462E+01  -0.10000E+01  -0.10000E+01


-- BEGIN ITERATION NUMBER    9

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (DF-VECTOR)
      1)   0.82166E-02   0.94493E-02   0.38751E-02

SEARCH DIRECTION (S-VECTOR)
      1)  -0.42778E+00  -0.10000E+01  -0.47765E+00

PROPOSED ALPHA =  0.22095E-01

CALCULATED ALPHA =  0.67373E-02

OBJECTIVE = 0.66144E-01
```

```
DECISION VARIABLES (X-VECTOR)
      1)  -0.66239E+00  -0.77353E+00  -0.62930E+01

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.37421E+00  -0.11589E+01  -0.89355E+01  -0.10000E+01  -0.10000E+01


-- BEGIN ITERATION NUMBER   10

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (DF-VECTOR)
      1)   0.78501E-03   0.27141E-02   0.34708E-02

SEARCH DIRECTION (S-VECTOR)
      1)  -0.22617E+00  -0.78196E+00  -0.10000E+01

PROPOSED ALPHA =  0.15427E-01

CALCULATED ALPHA =  0.21057E-02

OBJECTIVE = 0.66142E-01

DECISION VARIABLES (X-VECTOR)
      1)  -0.66287E+00  -0.77518E+00  -0.62951E+01

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.37304E+00  -0.11599E+01  -0.89333E+01  -0.10000E+01  -0.10000E+01


-- BEGIN ITERATION NUMBER   11

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (DF-VECTOR)
      1)  -0.86474E-03  -0.80495E-01   0.33776E-02

SEARCH DIRECTION (S-VECTOR)
      1)   0.10743E-01   0.10000E+01  -0.41960E-01

PROPOSED ALPHA =  0.44215E-02

CALCULATED ALPHA =  0.11576E-01

OBJECTIVE = 0.65590E-01

DECISION VARIABLES (X-VECTOR)
      1)  -0.66275E+00  -0.76360E+00  -0.62956E+01

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.39095E+00  -0.13935E+01  -0.89388E+01  -0.10000E+01  -0.10000E+01


-- BEGIN ITERATION NUMBER   12

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (DF-VECTOR)
      1)   0.92074E-01   0.21756E-01   0.12965E-01

SEARCH DIRECTION (S-VECTOR)
      1)  -0.99496E+00   0.10000E+01  -0.19393E+00
```

```
PROPOSED ALPHA =  0.68407E-02

CALCULATED ALPHA =  0.00000E+00

OBJECTIVE = 0.65590E-01

DECISION VARIABLES (X-VECTOR)
      1)  -0.66275E+00  -0.76360E+00  -0.62956E+01

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.39095E+00  -0.13935E+01  -0.89388E+01  -0.10000E+01  -0.10000E+01


-- BEGIN ITERATION NUMBER   13

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (OF-VECTOR)
      1)   0.92074E-01   0.21756E-01   0.12965E-01

SEARCH DIRECTION (S-VECTOR)
      1)  -0.10000E+01  -0.23629E+00  -0.14081E+00

PROPOSED ALPHA =  0.57878E-02

CALCULATED ALPHA =  0.10867E-03

OBJECTIVE = 0.65589E-01

DECISION VARIABLES (X-VECTOR)
      1)  -0.66285E+00  -0.76363E+00  -0.62956E+01

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.39071E+00  -0.13935E+01  -0.89385E+01  -0.10000E+01  -0.10000E+01


-- BEGIN ITERATION NUMBER   14

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

GRADIENT OF THE OBJECTIVE FUNCTION (OF-VECTOR)
      1)   0.92007E-01   0.21717E-01   0.12970E-01

SEARCH DIRECTION (S-VECTOR)
      1)  -0.10000E+01  -0.23616E+00  -0.14089E+00

PROPOSED ALPHA =  0.10000E-02

CALCULATED ALPHA =  0.72687E-04

OBJECTIVE = 0.65589E-01

DECISION VARIABLES (X-VECTOR)
      1)  -0.66293E+00  -0.76365E+00  -0.62956E+01

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.39054E+00  -0.13935E+01  -0.89383E+01  -0.10000E+01  -0.10000E+01



FINAL OPTIMIZATION RESULTS

NUMBER OF ITERATIONS =   14
```

```
OBJECTIVE = 0.65589E-01

DECISION VARIABLES (X-VECTOR)
      1)  -0.66293E+00  -0.76365E+00  -0.62956E+01

CONSTRAINT VALUES (G-VECTOR)
      1)  -0.39054E+00  -0.13935E+01  -0.89383E+01  -0.10000E+01  -0.10000E+01

CONSTRAINT TOLERANCE, CT = -0.30000E-01  CTL = -0.50000E-02

THERE ARE    0 ACTIVE CONSTRAINTS AND    0 VIOLATED CONSTRAINTS

THERE ARE    0 ACTIVE SIDE CONSTRAINTS

TERMINATION CRITERIA

RELATIVE CONVERGENCE CRITERION WAS MET FOR  3 CONSECUTIVE ITERATIONS

ABSOLUTE CONVERGENCE CRITERION WAS MET FOR  3 CONSECUTIVE ITERATIONS


                    ----------------------
                      OPTIMIZATION RESULTS
                    ----------------------


OBJECTIVE FUNCTION VALUE    0.65589E-01


DESIGN VARIABLES

                          LOWER                        UPPER
VARIABLE      BOUND         VALUE         BOUND
        1    -0.10000E+04  -0.66293E+00   0.10000E+04
        2    -0.10000E+04  -0.76365E+00   0.10000E+04
        3    -0.10000E+04  -0.62956E+01   0.10000E+04


DESIGN CONSTRAINTS

        1) -0.3905E+00 -0.1394E+01 -0.893BE+01 -0.1000E+01 -0.1000E+01

FUNCTION EVALUATIONS =    90
        *** DSL OUTPUT LISTING, GROUP  1                                    , STARTING RUN  92 ***
    TIME          OBJ           C           X1           X2           U
 0.00000E+00  0.00000E+00   1.0000      0.00000E+00  0.00000E+00  0.00000E+00
 1.00000E-02  1.90451E-02  0.99076      9.23799E-03  5.24768E-02  0.10900
 2.00000E-02  3.50469E-02  0.93704      6.29613E-02  0.16774      0.29362
 3.00000E-02  4.70222E-02  0.82258      0.17742      0.28635      0.51578
 4.00000E-02  5.51022E-02  0.65749      0.34251      0.36476      0.73190
 5.00000E-02  6.00777E-02  0.46817      0.53183      0.3B167      0.905B0
 6.00000E-02  6.28622E-02  0.28607      0.71393      0.33750       1.0164
 7.00000E-02  6.42180E-02  0.13799      0.B6201      0.24934       1.0604
 B.00000E-02  6.47264E-02  3.96980E-02  0.96030      0.14261       1.0494
 9.00000E-02  6.48304E-02 -5.97585E-03   1.0060      4.28098E-02   1.0042
 1.00000E-01  6.4B437E-02 -7.61740E-03   1.0076     -3.09017E-02  0.94839
 0.11000      6.49337E-02  1.89452E-02  0.9B105     -6.92305E-02  0.90176
 0.12000      6.51313E-02  5.58203E-02  0.94418     -7.29285E-02  0.87684
 0.13000      6.53820E-02  B.75761E-02  0.91242     -5.06554E-02  0.87737
 FINAL GAINS G(1),G(2),G(3) =     -0.6629E+00    -0.7636E+00    -0.6296E+01
 FINAL POLES ARE : (-0.1245E+02, 0.5156E+02 J)  (-0.1245E+02,-0.5156E+02 J)  (-0.2237E+02, 0.0000E+00 J)
SUMMARY:  ENDING RUN NO. =   92,  NO. OF RERUNS REQUESTED =   92
                *** RUN TERMINATED BY 'ENDJOB' ***
```

APPENDIX G

ADSLPC IMSL PROGRAM CHANGES

Changes were done to the following IMSL routines to make them compatible with MS-FORTRAN as follows:

Routine	*Line Number*	*Change*
DGEAR	4260	SEPS = 1.525878906E-05
	5980	6HDGEAR to 'DGEAR '
	5990	6HDGEAR to 'DGEAR '
LEQT1B	2190	6HLEQT1B to 'LEQT1B'
UERTST	530	define CHARACTER*6 NAME
	550	define CHARACTER IEQ
	560	define CHARACTER*6 NAMSET, NAMUPK
	570	NAMSET = 'UERSET'
	580	NAMEQ = ' '
	590	IEQ = '='
	620	NAMUPK = NAME
	640	IOUNIT = 6
	880	deleted
	890	IF (NAMUPK.NE.NAMSET) GOTO 25
	1050	6A1 to 1A6 (TWICE)
	1070	6A1 to 1A6 (TWICE)
	1090	6A1 to 1A6 (TWICE)
	1110	6A1 to 1A6 (TWICE)
	1170	deleted
	1180	NAMEQ=NAMUPK

EIGRF	1030	RDELP = 1.525878906E-05
	2740	6HEIGRF to 'EIGRF '
	2770	6HEIGRF to 'EIGRF '

EQRH3F	400	RDELP = 1.52878906E-05
	3700	6HEQRH3F' to 'EQRH3F'

Numerical conversions were calculated as follows:

- Original IBM data statement: DATA SEPS /Z3C100000/
- Find exponent: 3C - 40 = -3
- The resulting hexadecimal number is: 0.1×10^{-3}
- Converting to decimal the number is: 16.0×10^{-4}
- Converting to exponential form the value is: 1.525878906E-05

APPENDIX H

ADSLPC PROGRAM LISTING

1. ADSLPC MS-FOTRAN77 PROGRAM

```
$nofloatcalls
$large
$storage:2
C       DC MOTOR INTEGRAL CONTROL SYSTEM-NO NOISE-KUO EXAMPLE 8-7
C**********************************************************************
C
C LISTING OF ALL VARIABLES
C
C AA     MATRIX OF STATE COEFFICIENTS
C BB     VECTOR OF CONTROL COEFFICIENTS
C C      SCALAR OUTPUT VALUE
C CC     MATRIX OF STATE OUTPUT COEFFICEINTS
C CF     FINAL EXPECTED OUTPUT VALUE
C CI     INITIAL OUTPUT VALUE
C CON    VECTOR OF CONSTRAINT VALUES
C CSP    MAXIMUM PERCENT OVERSHOOT FOR THE OUTPUT
C DELC   DIFFERENCE OF THE ACTUAL OUTPUT TO THE INITIAL OUTPUT
C DELT   INITIAL INTEGRATION STEP SIZE
C DELU   DIFFERENCE OF THE ACTUAL CONTROL TO THE INITIAL CONTROL
C DELX2  DIFFERENCE OF THE ACTUAL X2 STATE TO THE INITIAL X2 STATE
C DV     VECTOR OF DESIGN VARIABLE GREADIENTS
C F      VECTOR OF INPUT COEFFICIENTS
C FV1    EIGENVALUE WORK VECTOR
C G      VECTOR OF DESIGN VARIABLE GAIN VALUES
C H      VECTOR OF OUTPUT COEFFICEINTS FOR THE INPUTS
C I      LOOP COUNTER
C II     EVALUATION COUNTER
C IC     VECTOR OF CONSTRAINT GRADIENTS
C IDG    VECTOR OF CONSTRAINT TYPES
```

```
C IERR   EIGENVALUE ERROR INDICATOR
C IGRAD  SCALAR FOR INDICATING METHOD FOR OBTAINING GRADIENTS
C INFO   SCALAR INDICATING OPTIMIZATION STATUS
C IONED  SCALAR INDICATING TYPE OF ONE-DIMESIONAL SEACH
C IOPT   SCALAR INDICATING TYPE OF OPTIMIZATION
C IPRINT SCALAR INDICATING TYPE OF PRINT CONTROL FOR ADS
C IFVT   LINPAC INTEGER WORK VECTOR
C ISTRAT SCALAR INDICATING TYPE OF STRATEGY
C IV1    EIGENVALUE INTEGER WORK VECTOR
C IWK    INTEGER WORK VECTOR FOR ADS
C J      LOOP COUNTER
C JOB    STATUS INDICATOR OF OPTIMIZATION PROBLEM
C LSLOPE SLOPE OF THE OBJECTIVE FUNCTION FOR THE LAST INTEGRATION
C LSTOBJ VALUE OF THE OBJECTIVE FUNCTION FOR THE LAST INTEGRATION
C LSTTIM VALUE OF TIME AT THE PREVIOUS INTEGRATION STEP
C MATZ   EIGENVALUE SCALAR VALUE TO INDICATOR EIGENVALUE DETERMINATION
C MEAN   DSL NORMAL DISTRIBUTION MEAN
C NCOLA  SCALAR COLUMN DIMENSION OF WA MATRIX
C NCON   NUMBER OF CONSTRAINTS
C NDV    NUMBER OF DESIGN VARIABLES
C NGT    SCALAR ADS CONTROL VARIABLE
C NM     DIMENSION OF MATRIX SIZE FOR EIGENVALUE DETERMIANTION
C NN     DIMENSION OF MATRIX SIZE FOR EIGENVALUE DETERMIANTION
C NRA    SCALAR ROW DIMENSION OF WA MATRIX
C NRIWK  SCALAR DIMENSION OF IWK VECTOR
C NRWK   SCALAR DIMENSION OF WK VECTOR
C OBJ    OBJECTIVE FUNCTION VALUE
C P      VECTOR OF DESIRED INITIAL COMPLEX POLE VALUES
C PA     MATRIX OF INITIAL GAIN DETERMINATION COEFFICIENTS
C Q      STATE WEIGHTING MATRIX
C R      CONTROL WEIGHTING VECTOR
C SD     STANDARD DEVIATION OF DSL NORMAL FUNCTION
C SEED   RANDOM MUMBER SEED FOR DSL NORMAL FUNCTION
C STEP   EVALUATION INTERVAL FOR CONSTRAINTS
C TAU    SYSTEM TIME CONSTANT
```

```
C TC      TIME WHEN COMPLETION TIME SPECIFICATION IS MET
C TIMEND TIMER VALUE AT EACH EVALUATION INTERVAL
C TOL     INITIAL TOLERANCE FOR DGEAR
C TSP     MAXIMUM COMPLETION TIME
C U       SCALAR FEEDBACK CONTROL VALUE
C UI      INITIAL VALUE OF THE CONTROL VECTOR
C UMAX    MAXIMUM CONTROL VALUE
C USP     MAXIMUM VALUE OF THE CONTROL VECTOR
C UT      TRANSLATED FEEDBACK CONTROL VALUE
C V       MATRIX FOR FINAL POLE DETERMINATION USING EIGENVALUE
C VLB     VECTOR OF DESIGN VARIABLE LOWER BOUNDS
C VUB     VECTOR OF DESIGN VARIABLE UPPER BOUNDS
C W1      DISTURBANCE VALUE
C W2      REFERENCE SETTING
C WA      ADS WORK VECTOR
C WL      CIRCULAR FREQUENCY OF SINUSOIDAL DISTRUBANCE
C WO      COMPLEX VECTOR OF FINAL POLES
C X1      MOTOR SHAFT SPEED STATE VALUE
C X1T     TRANSLATION OF X1 STATE
C X2      MOTOR CURRENT STATE VALUE
C X2I     INITIAL VALUE OF THE X2 STATE
C X2MAX   MAXIMUM VALUE OF THE X2 STATE
C X2SP    MAXIMUM SPECIFIED VALUE OF THE X2 STATE
C Y       VECTOR OF INTEGRATION FUNCTION VALUES
C YOSC1   VALUE OF SYSTEM OUTPUT OSCILATION AT TWO PREVIOUS INTEGRATIONS
C YOSC2   VALUE OF SYSTEM OUTPUT OSCILATION AT PREVIOUS INTEGRATION STEP
C YOSC3   VALUE OF SYSTEM OUTPUT OSCILATION
C YPRIME VECTOR OF INTEGRATION FUNCTION DERIVATIVES
C Z       EIGENVALUE DUMMY MATRIX
C**********************************************************************
C
          COMMON AA(3,3),BB(3),CC(3,3),H(2),R(2),Q(2,2),F(2,2),G(3),
         $ W1,W2,U
          INTEGER IDG(5),IC(20),IWK(400)
          REAL DV(3),VLB(3),VUB(3),CON(5),DF(3),WA(20,20),WK(400)
```

```
      REAL LSTTIM,LSTOBJ,LSLOPE
      COMPLEX P(3),WO(3),Z(3,3)
      INTEGER IGWK(4)
      REAL FV1(3),GWK(69),V(3,3),Y(4)
      EXTERNAL DUMMY,DERIVA
      OPEN(6,FILE='OUTPUT.DAT')
      OPEN(7,FILE='CON:')
C  * * * * * * * * *     INITIALIZATION SEGMENT     * * * * * * * * * * *
C INITIALIZE THE ADS PARAMETERS
      NRA=20
      NCOLA=20
      NRWK=400
      NRIWK=400
      NDV=3
      NCON=5
      IGRAD=0
      INFO=-2
C ESTABLISH TYPES OF CONSTRAINTS (ADS MANUAL TABLE 5)
      IDG(1)=-1
      IDG(2)=-1
      IDG(3)=-1
      IDG(4)=-1
      IDG(5)=2
C LOWER BOUND FOR DESIGN VARIABLES
      VUB(1)=1000.
      VUB(2)=1000.
      VUB(3)=1000.
C UPPER BOUND FOR DESIGN VARIABLES
      VLB(1)=-1000.
      VLB(2)=-1000.
      VLB(3)=-1000.
C ADS OPTIMIZATION METHOD SELECTION
      ISTRAT=1
      IOPT=3
      IONED=3
```

```
      IPRINT=1000
C*********************************************************************
C INITIALIZE THE INTEGRATION PARAMETERS
      FINTIM=1.0
      STEP=0.1
      TOL=1.E-3
C*********************************************************************
C SET CONSTRAINT SPECIFICATIONS
C CI=INITIAL OUTPUT VALUE;  CF=FINAL EXPECTED OUTPUT VALUE
C CSP=MAXIMUM PERCENT OVERSHOOT FOR THE OUTPUT
C TSP=MAXIMUM COMPLETION TIME
C USP=MAXIMUM VALUE OF THE CONTROL VECTOR
      CI=1.
      CF=0.
      CSP=100.
      TSP=1.5
      USP=10.0
      X2SP=10.0
C SET INITIAL INPUT VALUES (W1=DISTURBANCE, W2=REFERENCE SETTING)
      W1=0.
      W2=1.
      SEED=113.
      MEAN=1
      SD=0.1
      WL=157
C SET INITIAL RUN FLAG AND EIGENVALUE PARAMETERS
      MATZ=0
      NM=3
      NN=3
      II=0
C            * * * * * * * *  INITIAL        SEGMENT  * * * * * * * *
C SET ALL MATRIX AND VECTOR INPUTS
      P(1)=CMPLX(-10.0,10.0)
      P(2)=CMPLX(-10.0,-10.0)
      P(3)=CMPLX(-300.0,0.)
```

```
AA(1,1)=0.E0
AA(1,2)=50.E0
AA(1,3)=0.E0
AA(2,1)=-200.E0
AA(2,2)=-200.E0
AA(2,3)=0.E0
AA(3,1)=-1.E0
AA(3,2)=0.E0
AA(3,3)=0.E0
BB(1)=0.E0
BB(2)=200.E0
BB(3)=0.E0
CC(1,1)=-1.E0
CC(1,2)=0.E0
CC(1,3)=0.E0
CC(2,1)=0.E0
CC(2,2)=0.E0
CC(2,3)=0.E0
CC(3,1)=0.E0
CC(3,2)=0.E0
CC(3,3)=0.E0
H(1)=0.E0
H(2)=1.E0
F(1,1)=-50.E0
F(1,2)=0.E0
F(2,1)=0.E0
F(2,2)=0.E0
Q(1,1)=1.E0
Q(1,2)=0.E0
Q(2,1)=0.E0
Q(2,2)=1.E0
R(1)=1.E0
R(2)=0.E0
DV(1)=-1.
DV(2)=-1.
```

```
      DV(3)=-1.
      DO 610 I=1,NCON
  610 CON(I)=0.E0
C POLE TO GAIN ROUTINE
      CALL ADS(INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,DV,VLB,
     $ VUB,OBJ,CON,IDG,NGT,IC,DF,WA,NRA,NCOLA,WK,NRWK,IWK,NRIWK)
C TURN SCALING OFF
      IWK(2)=0
  660 CALL ADS(INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,DV,VLB,
     $ VUB,OBJ,CON,IDG,NGT,IC,DF,WA,NRA,NCOLA,WK,NRWK,IWK,NRIWK)
      IF(INFO.EQ.0)GOTO650
      DO 700 I=1,3
      G(I)=DV(I)
      DO 710 J=1,3
  710 V(J,I)=AA(J,I)-BB(J)*G(I)
  700 CONTINUE
C FIND EIGENVALUES FOR PROPOSED GAINS
      CALL EIGRF(V,NM,NN,MATZ,WO,Z,NM,FV1,IERR)
      RP1=REAL(P(1))
      RP2=REAL(P(2))
      RP3=REAL(P(3))
      RIP1=AIMAG(P(1))
      RIP2=AIMAG(P(2))
      RIP3=AIMAG(P(3))
      WR1=REAL(WO(1))
      WR2=REAL(WO(2))
      WR3=REAL(WO(3))
      WI1=AIMAG(WO(1))
      WI2=AIMAG(WO(2))
      WI3=AIMAG(WO(3))
      IF(AMAX1(WR1,WR2,WR3).GT.0.E0)G(1)=-G(1)
      CON(1)=ABS(AMAX1(WR1,WR2,WR3)-AMAX1(RP1,RP2,RP3))
      CON(2)=ABS(AMIN1(WR1,WR2,WR3)-AMIN1(RP1,RP2,RP3))
      CON(3)=ABS(AMAX1(WI1,WI2,WI3)-AMAX1(RIP1,RIP2,RIP3))
      CON(4)=ABS(AMIN1(WI1,WI2,WI3)-AMIN1(RIP1,RIP2,RIP3))
```

```
      OBJ=CON(1)+CON(2)+CON(3)+CON(4)
      GOTO 660
C PRINTOUT INITIAL POLES AND GAINS
  650 WRITE(6,296)P(1),P(2),P(3)
  296 FORMAT('0 DESIRED POLES: ',3('(',E10.4,',',E10.4,' J) '))
      WRITE(6,297)WR1,WI1,WR2,WI2,WR3,WI3
  297 FORMAT('0 INITIAL POLES: ',3('(',E10.4,',',E10.4,' J) '))
      WRITE(6,299)G(1),G(2),G(3)
  299 FORMAT('0 INITIAL GAINS G(1),G(2),G(3) = ',3E11.4)
      WRITE(6,290)
  290 FORMAT('0 STATE CONTROL MATRIX (B  AB) = ')
      WRITE(6,298)BB(1),AA(1,1)*BB(1)+AA(1,2)*BB(2)
      WRITE(6,298)BB(2),AA(2,1)*BB(1)+AA(2,2)*BB(2)
  298 FORMAT(2E15.4)
      WRITE(6,291)
  291 FORMAT('0 OUTPUT CONTROL MATRIX (CB  CAB) = ')
      WRITE(6,298)CC(1,1)*BB(1)+CC(1,2)*BB(2),CC(1,1)*(AA(1,1)*BB(1)+
     $ AA(1,2)*BB(2))+CC(1,2)*(AA(2,1)*BB(1)+AA(2,2)*BB(2))
      WRITE(6,292)
  292 FORMAT('0 OBSERVABILITY MATRIX (C*  A*C*) = ')
      WRITE(6,298)CC(1,1),AA(1,1)*CC(1,1)+AA(2,1)*CC(1,2)
      WRITE(6,298)CC(1,2),AA(1,2)*CC(1,1)+AA(2,2)*CC(1,2)
C CONTROLLER OPTIMIZATION ROUTINE
C RESET OPTIMIZATION PARAMETERS
      IWK(2)=1
      ISTRAT=0
      IOPT=4
      IONED=7
      IPRINT=3030
      IDG(1)=0
      IDG(2)=0
      IDG(3)=0
      IDG(4)=2
  200 CALL ADS(INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,DV,VLB,
     $ VUB,OBJ,CON,IDG,NGT,IC,DF,WA,NRA,NCOLA,WK,NRWK,IWK,NRIWK)
```

```
      IF(INFO.EQ.0)WRITE(6,300)
  300 FORMAT('     TIME          X1          X2          U ',
     $ '          OBJ         C')
C RESET CONSTRAINT AND RUN VALUES
      G(1)=DV(1)
      G(2)=DV(2)
      G(3)=DV(3)
      II=II+1
      JOB=0
      YOSC1=CI
      YOSC2=CI
      YOSC3=CI
      CMAX=0.E0
      TAU=0.E0
      TC=0.E0
      U=0.E0
      UI=0.E0
      UMAX=0.E0
      LSTOBJ=0.E0
      LSLOPE=0.E0
      LSTTIM=0.E0
C INITIALIZE THE DGEAR CONSTANTS FOR THE INTEGRATION TYPE DESIRED
      TIME=0.0E0
      TIMEND=0.E0
      DELT=0.001
      NDE=4
      METH=1
      MITER=0
      INDEX=1
      DO 201 I=1,NDE
  201 Y(I)=0.E0
C INCREMENT THE INTEGRATION INTERVAL
   20 TIMEND=TIMEND+STEP
      WRITE(7,455)II,TIME
  455 FORMAT(' EVAL #',I3,' TIME=',F5.3)
```

```
C CALL DIFFERENTIAL EQUATION SOLVER ROUTINE
      CALL DGEAR(NDE,DERIVA,DUMMY,TIME,DELT,Y,TIMEND,TOL,METH,
     $ MITER,INDEX,IGWK,GWK,IER)
C UPDATE STATE AND CONTROL VALUES
      X1=Y(1)
      X2=Y(2)
      OBJ=Y(4)
      C=CC(1,1)*X1+CC(1,2)*X2+H(1)*W1+H(2)*W2
      U=-G(1)*X1-G(2)*X2-G(3)*Y(3)
C            * * * * * * * *  DYNAMIC         SEGMENT  * * * * * * * *
C PRINT STATES AND OBJECTIVE - LAST RUN ONLY
      IF(INFO.EQ.0)WRITE(6,310)TIMEND,X1,X2,U,OBJ,C
  310 FORMAT(6G12.4)
C UPDATE CONSTRAINT CHECK VALUES
      YOSC1=YOSC2
      YOSC2=YOSC3
      YOSC3=C
      H1=H2
      H2=H3
      H3=ABS(YOSC3-CF)
      DELC=ABS(CI-C)
      DELU=ABS(UI-U)
C EVALUATE CONSTRAINTS
      IF(DELU.GE.UMAX)UMAX=DELU
      IF(TAU.NE.0.E0)GOTO 15
      IF(DELC.GE.ABS(0.63212E0*(CI-CF)))TAU=TIMEND
   15 IF(DELC.GE.CMAX)CMAX=DELC
      IF(DELC.EQ.CMAX)CON(1)=DELC-ABS(CI-CF)*(CSP/100+1.E0)
      IF(TC.NE.0.E0)GOTO 150
      IF((H2.GE.H1).AND.(H2.GE.H3).AND.(H2.LE.ABS((CI-CF)*.01E0)))
     $ TC=TIMEND
      IF((H3.LT.H2).AND.(H2.LT.H1).AND.(H1.LT.ABS((CI-CF)*.01E0)))
     $ TC=TIMEND
      IF((C.GT.(CF-ABS(CI-CF)*.01E0)).AND.(TIMEND.GE.(5.E0*TAU)).AND.
     $ (TAU.NE.0.E0).AND.(C.EQ.CMAX))TC=TIMEND
```

```
         CON(2)=10.E0
         IF(TC.NE.0.E0)CON(2)=TC-TSP
  150 IF(OELU.EQ.UMAX)CON(3)=OELU-USP
         CON(4)=-1.E0
         IF(ABS(C).GT.ABS(CI-CF)*10.E0*(CSP/100.E0+1.E0))CON(4)=1.E0
         IF(CON(4).GT.0.E0)JOB=1
         CON(5)=-1.E0
         IF((ABS(CF-C).GT.ABS(CF-CI)).AND.(TAU.EQ.0.E0))CON(5)=1.E0
         IF(CON(5).GT.0.E0)JOB=1
C EVALUATE FOR END OF RUN
         IF(TIMEND.LE.STEP*3.)GOTO 40
         SLOPE=ABS((OBJ-LSTOBJ)/(TIMEND-LSTTIM))
         DSLOPE=ABS((SLOPE-LSLOPE)/(TIMEND-LSTTIM))
         IF(((SLOPE.LT..001E0).OR.(DSLOPE.LE..001E0)).AND.(TC.NE.0.E0))
        $ JOB=1
   40 LSTOBJ=OBJ
         LSTTIM=TIMEND
         LSLOPE=SLOPE
C CHECK FOR END OF RUN CONDITION
         IF(JOB.EQ.1)GOTO 30
C CHECK FOR MAX TIME
         IF(TIMEND.GE.FINTIM)GOTO 30
         GOTO 20
C              * * * * * * * *  TERMINAL        SEGMENT  * * * * * * * *
C CHECK FOR END OF JOB
   30  IF(INFO.NE.0)GOTO 200
         DO 400 I=1,3
         DO 410 J=1,3
  410  V(I,J)=AA(I,J)-BB(I)*G(J)
  400  CONTINUE
C CALCULATE FINAL POLES
         CALL EIGRF(V,NM,NN,MATZ,WO,Z,NM,FV1,IERR)
         WRITE(6,497)WO(1),WO(2),WO(3)
         WRITE(6,499)G(1),G(2),G(3)
  497  FORMAT('0 FINAL POLES: ',3('(',E10.4,',',E10.4,' J) '))
```

```
  499  FORMAT('0 FINAL GAINS G(1),G(2),G(3) = ',3E11.4)
         STOP
         END
C              * * * * * * * *  DERIVATIVE      SEGMENT  * * * * * * * *
         SUBROUTINE DERIVA(NDE,TIME,Y,YPRIME)
         COMMON AA(3,3),BB(3),CC(3,3),H(2),R(2),Q(2,2),F(2,2),G(3),
        $W1,W2,U
         REAL Y(NDE),YPRIME(NDE)
         X1=Y(1)
         X2=Y(2)
C DIFFERENTIAL EQUATIONS FOR CONTROLLER
         X1DOT=AA(1,1)*X1+AA(1,2)*X2+BB(1)*U+F(1,1)*W1+F(1,2)*W2
         X2DOT=AA(2,1)*X1+AA(2,2)*X2+BB(2)*U+F(2,1)*W1+F(2,2)*W2
         Y3DOT=CC(1,1)*X1+CC(1,2)*X2+H(1)*W1+H(2)*W2
         U=-G(1)*X1-G(2)*X2-G(3)*Y(3)
         UT=U-1.E0
         X1T=X1-1.E0
         OBJDOT=(X1T*Q(1,1)+X2*Q(1,2))*X1T+(X1T*Q(2,1)+X2*Q(2,2))*X2+
        $ (UT*R(1)*UT)
         YPRIME(1)=X1DOT
         YPRIME(2)=X2DOT
         YPRIME(3)=Y3DOT
         YPRIME(4)=OBJDOT
         RETURN
         END
C DUMMY ROUTINE REQUIRED BY DGEAR
         SUBROUTINE DUMMY (N,X,Y,PD)
         INTEGER N
         REAL Y(N),PD(N,N),X
         RETURN
         END
```

2. ADSLPC BATCH FILE

```
echo off
rem: ASK is a program to query the user for a yes or no response
rem: KEDIT is the line editor program of choice
rem: PUTIME displays the current clock time
rem: PUTIMEB displays the current clock time and sounds the system bell
rem: BLANK.DATis an empty file
rem: SETPRN sends escapecodes to the printer
echo off
path  adsl util; adsl program: fortran; ; utiltys
cd adsl program
:start
ask Edit & Compile ADSLPC.FDR?
if errorlevel 1 goto start2
:begin
del ADSLPC.DBJ
:start3
KEDIT ADSLPC.FDR
putime
FOR1 ADSLPC.ADSLPC,NUL,NUL
putime
PAS2
if exist ADSLPC.DBJ goto link
putimeb
ask Re-edit & Compile ADSLPC.FDR?
if errorlevel 1 goto start2
goto start3
:start2
ask Link and Run the ADSLPC program?
if errorlevel 1 goto link2
:link
putime
LINK ADSLPC+ADS1A+ADS1B+ADS2A+ADS2B+ADS3A+ADS3B+ADS3C+IMSL1+IMSL2+IMSL3,MDDEL87,NUL, FDRTRAN 8087.LIB+ FORTRAN FDRTRAN.LIB
putime
goto run12
:link2
ask Run the ADSLPC model?
if errorlevel 1 goto print
:run12
copy blank.dat output.dat
putime
MDDELB7
putime
pause
BRDWSE output.dat
:print
ask Printout results (if yes--verify printer on, paper aligned)?
if errorlevel 1 goto done1
setprn [27]N[B]
setprn [15]
PRINT OUTPUT.DAT
ASK PRINT ADSLPC.FDR?
IF ERRDRLEVEL=1 GOTD NEXT2
PRINT ADSLPC.FDR
:NEXT2
:done1
ask Edit & Compile ADSLPC.FDR?
if errorlevel 1 goto done
goto begin
:done
echo on
```

LIST OF REFERENCES

1. Speckhart, F. H. and Green, W. L., *A Guide to Using CSMP*, Prentice-Hall, 1976.

2. International Business Machines Program Number 5798-PXJ, *Dynamic Simulation Language/VS Language Reference Manual*, June 1984.

3. Vanderplaats, G. N., *Numerical Optimization Techniques for Engineering Design*, McGraw-Hill, 1984

4. Vanderplaats, G. N., *ADS A Fortran Program for Automatic Design Synthesis, Version 1.10*, Engineering Design Optimization, Inc., 1985.

5. Ogata, K., *Modern Control Engineering*, Prentice-Hall, 1970.

6. Fletcher, R. and Reeves, C. M., "Function Minimization by Conjugate Gradients," *Br Computer J.*, v. 7, no. 2, 1964.

7. Kuo, B. C., *Automatic Control Systems*, 4th ed., Prentice-Hall, 1981.

8. Dongarra, J., et al, *LINPAC User's Guide*, Society for Industrial and Applied Mathematics, 1979.

9. Smith, B. T., et al, *Matrix Eigensystem Routines EISPACK Guide*, Second Edition, Springer-Verlag, 1976.

10. Nye, W. T. and Tits, A. L., "An Application-oriented, Optimization-based Methodology for Interactive Design of Eng. Systems," *International Journal of Controls*, v. 43, pp. 1693-1721, 1986.

11. Chow, W., *Automated Pole Placement Algorithm for Multivariable Optimal Control Synthesis*, MSAE Thesis, Naval Postgraduate School, Monterey, California, September 1985.

12. Gordon, V., *Utilization of Numerical Optimization Techniques in the Design of Robust Multi-Input, Multi-Output Control Systems*, PhD Thesis, Naval Postgraduate School, Monterey, California, September 1984.

13. Vanderplaats, G. N. and Sugimoto, H., *ADS Design Optimization Program*, FORTRAN Source Code, Naval Postgraduate School, Monterey, California, June 1982.

14. Microsoft Corporation, *MS-FORTRAN77 (Version 3.2) for MS-DOS Systems*, 1984.

15. Startz, R., *8087 Applications and Programing for the IBM PC, XT, and AT, Revised and Expanded*, Brady Communications, 1985.

16. IMSL Corporation, *International Mathematical and Statistical Library*, 1984.

17. NASA Technical Memorandum 87344, *HYTESSII- A Hypothetical Turbofan Engine Simplified Simulation with Multivariable Control and Sensor Analytical Redundancy*, by W. C. Merrill, June, 1986.

18. International Business Machines Program Number 5798-PXJ, *Dynamic Simulation Language/VS Operations Manual*, June 1984.

BIBLIOGRAPHY

Amin, M. H., "Optimal Discrete Systems with Prescribed Eigenvalues," *International Journal of Controls*, v. 40, pp. 783-794, 1984.

Athans, M. and Falb, P. L., *Optimal Control*, McGraw-Hill, 1966.

Bowles, R. J., *Sub-Optimal Control of a Gas Turbine Engine*, MS Thesis, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, December 1973.

Brogan, W. L., *Modern Control Theory*, Quantum, 1974.

Brown, G. D., *System 370 Job Control Language*, Wiley-Interscience, 1977.

Chintapalli, P. and Douglas, J. M., "The Use of Economic Performance Measures to Synthesize Optimal Control Systems," *Ind. Eng. Chem., Fundam*, v. 14, no. 1, pp. 1-10, 1975.

Davidson, E. and Furguson, I., "The Design of Controllers for the Multiv ariable Robust Servomechanism Problem," *Proceedings of the 19th IEEE Conference on Decision and Control*, IEEE Control System Society, Albuquerque, New Mexico, December 10-12, 1980.

Dundics, M. J., *CONSYN An Optimal Integral Output Controller Design and Analysis Program*, MSME Thesis, Naval Postgraduate School, Monterey, California, December 1977.

Feliachi, A. and Meliopoulos, A., "Computer Based Optimal Control Design Procedures in Large Scale Systems," *ASME Advances in Modeling and Simulation*, pp. 51-63, 1984.

Gorsline, G.W., *16-Bit Modern Microcomputers, the Intel I8086 Family*, Prentice-Hall, 1985.

Jensen, J. R., "A Case of a Linear Quadratic Optimal Design with a Weighting Matrix which is not Non-negative Definite," *International Journal of Control*, v. 39, pp. 1367-1373, 1984.

Jones, D. I. and Finch, J. W., "Comparison of Optimization Algorithms," *International Journal of Controls*, v. 40, pp. 747-761, 1984.

Katayama, T., and others, "Design of an Optimal Controller for a Discrete Time System," *International Journal of Controls*, v. 41, pp. 677-679, 1985.

Lunze, J., "The Design of Robust Feedback Controllers for Partly Unknown Systems by Optimal Control Procedures," *International Journal of Controls*, v. 36, pp. 611-630, 1982.

NASA Technical Paper 1876, *Method for Obtaining Reduced-Order Control Laws for High-Order Systems Using Optimization Techniques*, by V. Mukhopadhyag, and others, August 1981.

Oldenburger, R., *Optimal Control*, Holt, Rinehardt, and Winston, 1966.

Polak, E., and others, "Delight.MIMO: An Interactive Optimization Based Multivariable Control System Design Package," *Control Systems Magazine*, v. 2, pp. 9-14, December 1982.

Safiuddin, M., and Subba, V., *Microprocessor Based Control for Cyclic Duty DC Drives*, paper presented at the Industrial Applications Society Annual Meeting, October 3-7, 1983.

Sarma, I. G., and others, *Minimum Trajectory Sensitivity Design of Optimal Control Systems*, paper presented at the Third IFAC Symposium, Ischia, Italy, June 18-23, 1973.

Smith, H. W. and Davison, E. J., "Integral Feedback and Feed Forward Control," *Proc IEE*, vol 119, pp. 1210-1216, August 1972.

United Aircraft Research Laboratories Report No. Mg41338-2, *An Analytical Method for the Synthesis of Non-linear Multivariable Feedback Control*, by G. J. Michael and F. A. Farrar, 1973.

Vande Vegte, J., *Feedback Control Systems*, Prentice-Hall, 1986.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginina 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5002	2
3.	Chairman, Code 69Hy Mechanical Engineering Department Naval Postgraduate School Monterey, California 93943-5000	1
4.	Professor R. H. Nunn, Code 69Nn Mechanical Engineering Department Naval Postgraduate School Monterey, California 93943-5000	5
5.	Associate Professor D. L. Smith, Code 69Sm Mechanical Engineering Department Naval Postgraduate School Monterey, California 93943-5000	2
6.	Associate Professor D. Salinas, Code 69Sa Mechanical Engineering Department Naval Postgraduate School Monterey, California 93943-5000	2
7.	J. Favorite, Code 0141 W.R. Church Computer Center Naval Postgraduate School Monterey, California 93943-5000	2
8.	Dr. G.N. Vanderplaats Engineering Design Optimization, Inc. 1275 Camino Rio Verde Sanat Barbara, California 93111	2
9.	LCDR T. F. Olson, USN RT 2, Box 74 Stanfordville, NY 12581	5

Thesis
05443
c.1

221144

Olson
Application of numerical optimization in modern control.